Latent Space Model for Higher-Order Networks and Generalized Tensor Decomposition

Zhongyuan Lyu\textsuperscript{a}, Dong Xia\textsuperscript{a}, and Yuan Zhang\textsuperscript{b}

\textsuperscript{a}Department of Mathematics, Hong Kong University of Science and Technology, Kowloon, Hong Kong; \textsuperscript{b}Department of Statistics, Ohio State University, Columbus, OH

1. Introduction

Networks (Newman 2018) capturing the dyadic or pairwise interactions between a set of entities/vertices have been an active research field for more than half a century, leading to millions\textsuperscript{1} of publications and technical reports in related disciplines and a wide spectrum of applications. To date, various aspects of networks, for example, fundamental theories, statistical models, efficient algorithms and so forth, have been well developed through joint contributions from distinct scientific communities—physics, computer science, mathematics, statistics, to name a few. However, the recent decade has witnessed a fast growing demand in processing and analyzing more complex networks where interactions among a set of entities are polyadic or nonlinear. These complex networks pose fresh challenges on understanding and exploiting the joint interactions among entities.

The recent boom in data science gives rise to numerous categories of complex networks where relations among entities are far beyond being dyadic. More concretely, we focus on three specific types of complex networks—multi-layer networks (Kivelä et al. 2014), hypergraph networks (Ghoshal et al. 2009) and dynamic/temporal networks (Goldenberg et al. 2010), each of which is an independent subfield of study and has tremendous applications. Multi-layer networks arise when two vertices can present multiple types of relations, for instance, friendship networks (Dickison, Magnani, and Rossi 2016; Wang and Li 2020) on LinkedIn, Instagram, and Facebook among the same set of people can differ drastically; trading patterns of different commodities (Jing et al. 2021; Cai, Li, and Xia 2021) among the same set of countries are distinct. Other notable examples of multi-layer networks include brain fMRI images (Tang et al. 2017; Le, Levin, and Levina 2018; Wang, Zhang, and Dunson 2019; Paul and Chen 2020a; Arroyo et al. 2021), genetic networks and protein–protein interaction networks (Larremore, Clauset, and Buckee 2013; Hore et al. 2016; Zhang and Cao 2017; Lei, Chen, and Lynch 2020), transportation networks (Cardillo et al. 2013a, 2013b) and etc. Note that the interactions (node $i$, node $j$) in multi-layer networks on each layer are still dyadic. But they can be viewed as polyadic interactions (node $i$, node $j$, layer $l$) if layers are treated as an independent set of entities. Hypergraph networks refer to the complex systems whose vertex interactions are representable by hypergraphs consisting of a set of vertices and a set of hyper-edges. Each hyper-edge can connect multiple (more than 2) vertices exhibiting a polyadic relationship among these vertices, say (node $i$, node $j$, node $k$). A hypergraph is said to be $m$-uniform if every hyper-edge connects exactly $m$ vertices. Unlike the pairwise interaction of an edge, a hyper-edge captures the higher-order interaction which

\textsuperscript{1}Google Scholar reports ~ 5.8 million results to the search query “network analysis”.

© 2023 American Statistical Association and Institute of Mathematical Statistics
often carries more insightful information. In Benson, Gleich, and Leskovec (2016), the authors discover that, by incorporating high-order interactions in the airport network, the spectral clustering algorithm reveals geographic proximity between airports which is unseen if only dyadic relationships are used. Hypergraph networks are typically observed in co-authorship networks (Newman 2011; Ji and Jin 2016; Cai, Li, and Xia 2021), legislator network (Lee, Magallanes, and Porter 2017; Ke, Shi, and Xia 2019), proton emission networks (Zhen and Wang 2021), circuit networks (Ghoshdastidar and Dukkipati 2014) and so on. Lastly, dynamic/temporal networks (Wang, Yu, and Rinaldo 2018) naturally model dynamic systems where interactions between the same set of vertices evolve through time. They resemble multi-layer networks in the sense that layers are now indexed in a meaningful order, such as a discrete time flow. At a fixed time point, the relationship between vertices is still pairwise. Clearly, dynamic networks can be treated as networks of polyadic interactions, say (node $i$, node $j$, timestamp $t$), if the discrete time flow is viewed as a separate set of entities. Typical examples of dynamic networks include, for instance, the Senate cosponsorship network (Wang et al. 2017), Enron email network (Park, Priebe, and Youssef 2012; Wang et al. 2013), social interactions between animals (Matias and Miele 2017) and student friendship network (Chen and Zhang 2015).

The goal of this article is to investigate the aforementioned complex and unweighted networks—(mixture) multi-layer networks, hypergraph networks and dynamic/temporal networks in a unified framework. Since these networks all involve joint interactions of multiple entities, we collectively refer to these networks as higher-order networks. We note that, during the preparation of this work, the same concept was also coined by Bick et al. (2021).

Stochastic block model (SBM) (Holland, Laskey, and Leinhardt 1983) is a prevalent approach for modeling the latent group structures of vertices in networks. At the core of SBM is the assumption that vertices belonging to the same group are stochastically equivalent. The group structure of SBM intrinsically impose low-rank constraint on the expected adjacency matrix which naturally popularizes the spectral methods (Rohe, Chatterjee, and Yu 2011; Lei and Rinaldo 2015; Zhang, Levina, and Zhu 2016, 2020). Undoubtedly, numerous variants of SBM have been proposed to treat higher-order networks. The multi-layer SBM was proposed in Lei, Chen, and Lynch (2020), Paul and Chen (2020b), and Arroyo et al. (2021) assuming the same group assignments across all layers. A random effect multi-layer SBM was proposed in Paul and Chen (2020a) allowing for heterogeneous group assignments for different layers. More recently, Jing et al. (2021) introduced a novel mixture multi-layer SBM to simultaneously cluster networks and identify global and local group memberships of vertices. Among these prior works, the low-rankness of adjacency matrix and tensor is the primary ingredient in their methods. Similarly, hypergraph SBM was proposed and theoretically investigated in Ghoshdastidar and Dukkipati (2015), Ghoshdastidar and Dukkipati (2017), Chien, Lin, and Wang (2018), Kim, Bandeira, and Goemans (2018), Pal and Zhu (2019), and Yuan et al. (2018), where the expected adjacency tensor admits a low-rank decomposition. Meanwhile, Ke, Shi, and Xia (2019) introduced a degree corrected hypergraph SBM to accommodate the degree heterogeneity commonly observed in practice. The authors also proposed a low-rank tensor-based spectral method for community detection. For modeling the group structures in dynamic networks, SBM is also much favored. For instance, Pensky (2019), and Pensky and Zhang (2019) studied a dynamic SBM model and a spectral method for community detection. Aside from vertices clustering, another practically relevant problem in dynamic SBM is to detect change points in the sense that, for example, when network structure suddenly shifts. See, for example, Park, Priebe, and Youssef (2012), Wang et al. (2013), Wilson, Stevens, and Woodall (2019), Wang et al. (2017), and Wang, Yu, and Rinaldo (2018) for more details. All the aforementioned SBM extensions were designed for treating high-order networks. Without loss of generality, we will collectively refer to them as the high-order SBM.

Higher-order SBM enjoys structural simplicity, motivates diverse new statistical methods and demonstrates effective performances in identifying clusters. However, the stringent model assumptions of SBM may hamper or even jeopardize its effectiveness in handling more general higher-order networks. First of all, SBM enforces transitivity ($i$ connects to $j$, $j$ connects to $k$ implies $i$ connects to $k$ with high probability) via the cluster structure, that is, nodes in the same cluster tend more likely to connect. However, such strong clustering phenomenon may not be prevalent, especially in high-order networks. Recent advances in analyzing multi-layer networks, such as change point detection in dynamic networks, no longer limit themselves to block model structures (Wang, Yu, and Rinaldo 2018). Second, higher-order SBM usually makes the impractical assumption that nodes in the same cluster are stochastically equivalent. As an example, the trading flows of commodities between countries in Section 6; even though China, Germany, and United States share similar trading patterns of industrial commodities with other countries, and are identified as being close by a clustering algorithm, they clearly should not be regarded as equivalent in view of the striking technological gaps between these three economies. Finally, due to the linear relations, higher-order SBM usually results into an expected adjacency tensor admitting a low-rank decomposition (Ke, Shi, and Xia 2019; Jing et al. 2021). Unfortunately, oftentimes, the observed adjacency tensor presents many moderate-magnitude singular values rendering the low-rank presumption questionable.

As argued in Hoff, Raftery, and Handcock (2002), the transitivity of relations in networks may be better characterized by the proximity between vertices in an unobserved latent space, where each entity/vertex is associated with a vector of characteristics, named latent position, in this space. It is therefore referred to as the latent space model (LSM). Compared with SBM, the learned latent features from LSM (Levin et al. 2017; Ma, Ma, and Yuan 2020; MacDonald, Levina, and Zhu 2020; Zhang, Xue, and Zhu 2020) is sometimes more useful in downstream tasks such as node visualization, link prediction and community detection. Meanwhile, LSM allows for nonlinear relations with a general link or kernel function. In this article, we propose a unified framework based on LSM to treat higher-order networks—thus, the name higher-order latent space model (hLSM). Without loss of generality, we focus on higher-order networks with triadic
interactions among vertices. Let \( V_1, V_2, V_3 \) be three sets of “vertices” so that a triadic interaction of vertices \( i_1 \in V_1, i_2 \in V_2, i_3 \in V_3 \) is notionally regarded as a tuple \((i_1, i_2, i_3)\). We emphasize the abstraction of “vertices” in our framework since they can stand for conceptually different subjects in different contexts. In a hypergraph network, \( V_1, V_2, V_3 \) are the same set of vertices and the tuple \((i_1, i_2, i_3)\) just represents a hyper-edge connecting the three vertices. For a multi-layer or dynamic network, \( V_1 \) and \( V_2 \) can be the same set of vertices while \( V_3 \) is viewed as the index set of layers or time-stamps, respectively. Underlying our hLSM is the major assumption that each vertex \( i_k \in V_k \) for \( k = 1, 2, 3 \), is associated with a latent position in a low-dimensional space \( \mathcal{X}_i \). Conditioning on the latent positions, hLSM assumes that three vertices positioned \( u^*_i \in \mathbb{R}^{V_1}, \hat{v}^*_i \in \mathbb{R}^{V_2}, w^*_i \in \mathbb{R}^{V_3} \) would form triadic interaction \((i_1, i_2, i_3)\), independently of others, with probability \( \rho(u^*_i, \hat{v}^*_i, w^*_i) = g(C^*, u^*_i \otimes \hat{v}^*_i \otimes w^*_i) \), where \( g(\cdot) \) is a known link function and \( C^* \) is an unknown interaction tensor. Under the independent-edge assumption, the adjacency tensor \( [A]_{i_1,i_2,i_3} \) is Bernoulli\((g((\Theta^*)_{i_1,i_2,i_3}))\) for an unknown low-rank tensor \( \Theta^* = C^* \cdot [U^*, V^*, W^*] \). Here \([,] \) represents multilinear product, see formal definition in the last paragraph of this section. We estimate the latent positions \( U^*, V^*, W^* \) via the maximum likelihood estimator which is formulated as a problem of generalized low-rank tensor decomposition. Unfortunately, the objective function is highly nonconvex and can be solved only locally. Due to the orthogonality assumptions, the latent positions can be treated as points on Grassmann manifolds. We then propose a projected gradient descent algorithm on Grassmannians. Moreover, we also characterize the statistical error of the final estimates of latent positions for general high-order LSM’s. The error rate, determined by the signal strength of interaction tensor and the smoothness of the link function, is optimal in terms of the degrees of freedom. These results are applicable to a novel mixture multi-layer latent space model (MMLSM) and the hypergraph latent space model (hyper-LSM) since they are special cases under our general framework. In particular, our framework is capable of detecting heterogeneous latent positions in multi-layer networks and cluster the layers of networks which might admit similar latent positions. Finally, we also apply our method to a simple dynamic latent space model for change point detection.

Our main contributions can be summarized as follows. First, we introduce a general latent space model, called hLSM in short, to characterize polyadic interactions in higher-order networks, where the participating entities can be real actors in networks or virtual “vertices.” Second, in order to treat heterogeneous multi-layer networks, we propose a novel mixture multi-layer LSM. Unlike the existing literature on multi-layer LSM, our model allows distinct latent positions across layers, prevalent in many real-world applications. Other special cases of hLSM, including hypergraph LSM and dynamic LSM, are presented as well. Third, we formulate a general framework to estimate the latent positions by the maximum likelihood estimator, and propose a projected gradient descent algorithm on Grassmannians. We prove that the algorithm converges linearly if initialized well, and establish the statistical error of final estimates for both general and specific hLSM’s. Finally, the effectiveness of our algorithm is validated on comprehensive simulations and two real-world datasets. We showcase the merits of latent space models in the tasks of node embedding and link prediction.

Notation and Preliminaries on Tensors. Throughout the article, we use \( c, c_0, c_1, \ldots \) and \( C, C_0, C_1, \ldots \) to denote small and large absolute and positive constants, respectively. We write \( x \asymp y \) indicating that positive \( x \) and \( y \) are of same order, that is, \( cy \leq x \leq Cy \). Denote \( e_j \) the \( j \)-th canonical base vector whose dimension might vary, depending on the context. For an integer \( m \), denote \([m] := \{1, \ldots, m\}\). Let \( \mathbb{S}_n = \{X \in \mathbb{R}^{n \times p} : X^T X = I_{p \times p}\} \) be the collection of all column-orthonormal \( n \times p \) matrices. We use uppercase fonts, for example, \( U, W, \) to denote matrices and bold uppercase fonts, for example, \( A, \Theta, \) for tensors. Denote the \((i,j,k)\)th entry of a matrix \( A \) by \( [A]_{i,j,k} \). For any matrix \( A \) with rank \( \text{rank}(A) = r \), let \( \sigma_1(A) \geq \sigma_2(A) \geq \cdots \geq \sigma_r(A) > 0 \) denote its nonzero singular values. Define \( \sigma_{\max}(A) := \sigma_1(A) \) and \( \sigma_{\min}(A) := \sigma_r(A) \). Denote \( \|A\|_F, \|A\|_{\infty} \) the spectral norm and max norm of the matrix \( A \), respectively. We write \( \|A\|_2 := \max_j \|e_j^T A\| \).

For an \( n_1 \times n_2 \times n_3 \) tensor \( A \), its first matricization (also called unfolding) \( M_1(A) \in \mathbb{R}^{n_1 \times (n_2 n_3)} \) is defined by \([M_1(A)]_{1,i_2,i_3} = [A]_{i_1,i_2,i_3} \) for all \( i_1 \in [n_1] \). The second and third matricization of \( A \) are defined in a similar fashion. The Tucker ranks of \( A \) are defined by \( \text{rank}(A) = (\text{rank}(M_1(A)), (\text{rank}(M_2(A)), (\text{rank}(M_3(A)))) \). Given a matrix \( T \in \mathbb{R}^{n_1 \times n_2} \), the multi-linear product, denoted by \( \times_1 \), between \( A \) and \( U \) is defined by \([A \times_1 U]_{i_1,i_2} = \sum_{j=1}^{n_3} [A]_{i_1,i_2,j} [U]_{i_3,j} \) for \( i_1 \in [n_1], i_2 \in [n_2] \) and \( i_3 \in [n_3] \). The other multi-linear
products $x_2$ and $x_3$ are defined similarly. If $A$ has Tucker ranks $(r_1, r_2, r_3)$, there exists an $r_1 \times r_2 \times r_3$ tensor $C$, $U \in \mathbb{O}_{n_1, r_1}$, $V \in \mathbb{O}_{n_2, r_2}$, and $W \in \mathbb{O}_{n_3, r_3}$ such that

$$A = C \cdot [U, V, W] := C \times_1 U \times_2 V \times_3 W,$$ (1)

where for any $i_1 \in [n_1], i_3 \in [n_2], i_3 \in [n_3],$

$$[C \times_1 U \times_2 V \times_3 W]_{i_1 i_2 i_3} = \sum_{l_1=1}^{r_1} \sum_{l_2=1}^{r_2} \sum_{l_3=1}^{r_3} [C]_{l_1 l_2 l_3} [U]_{i_1 l_1} [V]_{i_2 l_2} [W]_{i_3 l_3}.$$

This is often referred to as the Tucker decomposition of $A$. We use $\Lambda(A) := \max \{\|M_k(A)\|, k = 1, 2, 3\}$ and $\Delta(A) := \min \{\sigma_{\min}(M_k(A)), k = 1, 2, 3\}$ to denote the largest and smallest singular values of the tensor $A$. For more details, please refer to Cai and Zhang (2018), Kolda (2006), and Kolda and Bader (2009).

2. Higher-Order Latent Space Model

For ease of exposition, we only present the hLSM for third-order networks, that is, all interactions among “vertices” are triadic. Its extension to higher-order ($\geq 3$) networks is conceptually straightforward. Without loss of generality, consider that there exist three sets of “vertices” $V_1$, $V_2$, and $V_3$ with size $n_k = |V_k|$. Here “vertices” are abstractions of “actors” in higher-order networks that can stand for even virtual subjects such as the index of layers in multi-layer networks and time-stamps in dynamic networks.

The observed third-order network is denoted by $G = (V, E)$ with a set of vertices $V = \{V_1, V_2, V_3\}$ and a set of triadic interactions $E$. A triadic interaction is a tuple $(i_1, i_2, i_3)$ with vertex $i_k \in V_k$. We say the triadic interaction among the vertices $i_1, i_2, i_3$ occurs if $(i_1, i_2, i_3) \in E$. The occurrences of distinct triadic interactions are assumed independent akin to the independent-edge random hypergraph (Ke, Shi, and Xia 2019). In hLSM, each vertex is associated with a latent position in an unobserved low-dimensional space characterizing inherent natures of the subjects, for example, the latent factor for the conservative versus liberal political ideology of senators (Chen, Li, and Xu 2021). For any tuple $(i_1, i_2, i_3)$, let $u^*_i \in \mathbb{R}^n, v^*_i \in \mathbb{R}^2$ and $w^*_i \in \mathbb{R}^3$ be the latent positions of these vertices. Here $r_k$ denotes the dimension of the latent space and it usually does not grow as the network size increases, for instance, the political ideology of a senator can be described by a 2-dim vector—conservatism versus liberalism. Nevertheless, our framework still applies to the cases where $r_k$ grows with the network size.

We introduce a kernel function $\rho(\cdot) : \mathbb{R}^n \times \mathbb{R}^2 \times \mathbb{R}^3 \mapsto [0, 1]$ such that the triadic interaction $(i_1, i_2, i_3)$ is generated with probability $\rho(u^*_i, v^*_i, w^*_i)$. Fixing the kernel function, the connection probability is determined solely by the latent positions. Denote $A \in \{0, 1\}^{n_1 \times n_2 \times n_3}$ the binary adjacency tensor of $G$ whose entries are $[A]_{i_1 i_2 i_3} = 1((i_1, i_2, i_3) \in E)$. Under hLSM, we have

$$[A]_{i_1 i_2 i_3} \overset{\text{ind}}{\sim} \text{Bernoulli}(\rho(u^*_i, v^*_i, w^*_i)), \quad \forall i_k \in V_k$$ (2)

Denote $U^* = [u^*_1, \ldots, u^*_n] \in \mathbb{R}^{n_1 \times n_1}$ (also $V^*, W^*$ resp.) the collection of all latent positions of $V_1$ (also $V_2, V_3$, resp.). By observing the adjacency tensor $A$ obeying (2), our goal is to estimate the latent positions $U^*, V^*$, and $W^*$.

The general class of kernel functions is too large to estimate. For simplicity, we assume that $\rho(\cdot)$ is a generalized multi-linear function in the sense that

$$\rho(u^*, v^*, w^*) = g((C^*, u^* \otimes v^* \otimes w^*))$$

$$= g \left( \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \sum_{i_3=1}^{n_3} [C^*]_{i_1 i_2 i_3} u^*_{i_1} v^*_{i_2} w^*_{i_3} \right)$$ (3)

where $C^* \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is an unknown parameter, called the interaction tensor, to be estimated. Here $\otimes$ denotes tensor product and $(\cdot, \cdot)$ is the Euclidean inner product. The function $g(\cdot)$ is a known link function, for instance the logistic function $g(x) = (1 + e^{-x})^{-1}$ and the probit function $g(x) = \Phi(x)$ where $\Phi(\cdot)$ is the cdf of standard normal random variable. With (2) and (3), we write the expected adjacency tensor by $E A = g(C^* \cdot [U^*, V^*, W^*])$, where we slightly abuse the notation and let $g(\cdot) : \mathbb{R} \mapsto \mathbb{R}$ also apply entry-wise on a tensor. If $g(x) = x$ and the latent positions have cluster structures, the model reduces to a higher-order SBM where the expected adjacency tensor admits a low-rank decomposition. Under hLSM with a general link function, $E A = g$ can be full rank while $g^{-1}(E A)$ is low-rank. Denote $\Theta^* = C^* \cdot [U^*, V^*, W^*]$ and then

$$[A]_{i_1 i_2 i_3} \overset{\text{ind}}{\sim} \text{Bernoulli}(g(\Theta^*))_{i_1 i_2 i_3}, \quad \forall i_k \in V_k.$$ (4)

Note that the independence of entries might hold only for a subset of all entries, for example, the off-diagonal entries for undirected graphs. Clearly, $\Theta^*$ can be uniquely determined by $E A$ if the function $g(\cdot)$ is monotonic. However, the latent positions are unidentifiable even with a given $\Theta^*$. Without loss of generality, we assume orthonormal latent positions so that $n_1^{-1} U^* U^*, n_2^{-1} V^* V^*$ and $n_3^{-1} W^* W^*$ are all identity matrices. Although similar conditions have appeared in Zhang, Xue, and Zhu (2020) and Zhang, Xue, and Zhu (2021), they are nonessential to our framework but only ease the algorithm design. We remark that the latent positions sometimes can possess additional structural properties, among which the incoherence is the most prevailing (Ke, Shi, and Xia 2019; Han, Willett, and Zhang 2020; Cai, Li, and Xia 2021; Jing et al. 2021). The incoherence constant of latent position $U^*$ is defined by

$$\text{Incoh}(U^*) := n_1^{1/2} \parallel U^* \parallel_F^{-1} \cdot \max_{1 \leq j \leq n_1} \|e_j^T U^*\|$$ (5)

Basically, if $\text{Incoh}(U^*)$ is upped bounded by a constant, it implies that the majority rows of $U^*$ have comparable and small magnitudes. It also means that the information $\Theta^*$ carries is fairly spread over all its entries.

For ease of references, we refer to hLSM($C^*, U^*, V^*, W^*, g(\cdot)$) as the higher-order LSM with parameters $C^* \cdot [U^*, V^*, W^*]$ and link function $g(\cdot)$. We now illuminate specific examples of hLSM for mixture multi-layer networks, hypergraph networks and dynamic networks.

2.1. Mixture Multi-Layer Latent Space Model

A multi-layer network often consists of multiple networks on the same set of vertices. Denote by $G = (\mathcal{V}, \cup_{j=1}^L \mathcal{E}_j)$ a multi-layer network that is composed of $L$ layers on the set of vertices $\mathcal{V}$ of size $|\mathcal{V}| = n$. The $l$th layer of network, denoted by $G_l = (\mathcal{V}, \mathcal{E}_l)$, is
an undirected binary graph. This is a special third-order network with $V_1 = V_2 = V$ and $V_3 = [L]$, and $n_1 = n_2 = n, n_3 = L$. Then, its adjacency tensor $A \in \{0, 1\}^{n \times n \times L}$ with $l$th slice $[A]_l$ being the adjacency matrix of the $l$th layer.

In Zhang, Xue, and Zhu (2020), the authors introduced a multi-layer LSM assuming the unchanged latent positions of vertices across all layers. However, in practice, similarities between vertices can shift drastically on different layers. For instance, when trading industrial commodities with other countries, China and United States are quite similar; whereas these two countries are in completely different positions when trading natural products with other countries. This suggests that a more reasonable model should allow heterogeneous latent positions across different layers. Toward that end, we propose a novel generative model, called mixture multi-layer latent space model (MMLSM). It can be regarded as a generalization of the mixture multi-layer SBM (Jing et al. 2021).

Suppose that there exists a mixture of $m$ LSMs and each layer $G_l$ is independently sampled from one of these LSMs. Now each layer has a latent label indicating which class of LSM it is sampled from. More specifically, for each $j \in [m]$, the $j$th class LSM is described by the latent positions $U_j \in \mathbb{R}^{n \times q_j}$ with $n^{-1}F_{U_j}U_j$ being identity and by a $q_j \times q_j$ interaction matrix $C_j$. Given a link function $g(\cdot)$, if $G_l$ is sampled from the $j$th class LSM, its expected adjacency matrix is simply $g(U_jC_jU_j^\top)$. For simplicity, we denote

- LSM($U_j, C_j, g(\cdot)$) — the $j$th class LSM with parameter $U_j, C_j$ and link function $g(\cdot)$.
- $s_l \in [m]$ — the latent label of $l$th layer for any $l \in [L]$. Denote $\mathcal{S} = \{s_1, \ldots, s_L\}$.
- $L = \#\{l : s_l = j, l \in [L]\}$ — the number of layers generated by the $j$th class LSM.

Throughout this article, we regard the layer labels $\mathcal{S}$ as being fixed. Consequently, the observed adjacency tensor obeys

$$[A]_{i_1,i_2,l}^{\text{ind}} \overset{\text{Bernoulli}}{\sim} \text{Bernoulli}\left(\left\{ g(U_jC_jU_j^\top) \right\}_{j \in \mathcal{S}} \right), \quad \forall (i_1, i_2, l) \in [n] \times [n] \times [L].$$

We call $U_j$ the local latent positions of the $j$th class LSM. Vertices $i_1$ and $i_2$ are locally similar in the $j$th class LSM if the $i_1$th and $i_2$th rows of $U_j$ are close. Let $\hat{U} = (U_1, \ldots, U_m) \in \mathbb{R}^{n \times q}$ be the collection of all local latent positions where $\hat{q} = \sum_{j = 1}^{m} q_j$. The closeness between the $i_1$th and $i_2$th row of $\hat{U}$ implies the global similarities of vertices $i_1$ and $i_2$ across all layers. MMLSM can be written in the form of hLSM. Define the $\hat{q} \times \hat{q} \times m$ interaction tensor $C$ such that its $j$th slice $C_{\cdot \cdot j}$ equals $[q_0, \ldots, q_{i_1}, C_j, q_{i_2}, \ldots, q_m]$, where $q_0$ denotes the $q \times q$ all-zero matrix. Denote the $L \times m$ layer-label matrix $W = (w_1, \ldots, w_m)^\top$ with $w_l$ being the $l$th canonical basis vector in $\mathbb{R}^m$. Thus, we can write $\Theta^* = C \cdot \hat{U}, \hat{W}$ and $\mathbb{E}A = g(\Theta^*)$.

Let $W^* = L^{1/2} \cdot W \cdot \text{diag}(L^{-1/2}, \ldots, L^{-1/2})^\top$ be the layer latent position matrix such that $L^{-1/2} \cdot W^* \cdot L^{-1/2} = \hat{U}^*$. The latent position $W^*$ reflects how layer label, as an independent “actor,” affects vertex interactions. But $\hat{U}$ may be rank deficient and thus inappropriate to be treated as global latent positions. Denote $r = \text{rank}(\hat{U})$ and $n^{-1/2}U^*$ the top-$r$ left singular vectors of $\hat{U}$ so that $n^{-1/2}U^*U^*$ is the identity matrix. We refer to $\hat{U}^*$ as the global latent positions of vertices. Therefore, $\Theta^*$ can be reparameterized and written as $\Theta^* = C \cdot \left[ \hat{U}^*, \hat{U}, W^* \right]$ where the new interaction tensor $C^*$ is of size $r \times r \times m$. Clearly, $nL^{1/2}C^*$ is attainable by multiplying $C$ with singular values and right singular vectors of $\hat{U}$ in the 1st and 2nd modes, and with $\text{diag}(L_1^{1/2}, \ldots, L_m^{1/2})$ in the third mode, accordingly. Finally, we write

$$[A]_{i_1,i_2,l}^{\text{ind}} \overset{\text{Bernoulli}}{\sim} \text{Bernoulli}\left(\left\{ g(C^* \cdot \left[ \hat{U}^*, \hat{U}, W^* \right]) \right\}_{l \in [L]} \right),$$

implies that the MMLSM is an hLSM with parameters $C^*, \hat{U}^*, W^*$ and the link function $g(\cdot)$. In MMLSM, we aim to estimate the local latent positions $U_j$’s, layer latent positions $W^*$ and global latent positions $\hat{U}^*$.

We remark that, although we focus on undirected networks, there is no substantial difficulty to generalize our framework to directed cases, in which the entries of parameter tensor can be written in the form $\Theta^* = C^* \cdot \left[ \hat{U}^*, \hat{V}^*, W^* \right]$.

### 2.2. Hypergraph Latent Space Model

A hypergraph network models higher-order interactions, called hyperedges, among a set of vertices. Without loss of generality, we focus on 3-uniform hypergraph where each hyperedge connects exactly three vertices. We now propose the hypergraph latent space model (hyper-LSM). Let $G = (V, E)$ be a 3-uniform undirected binary hypergraph with $V = [n]$ being the set of vertices and $E$ being the set of hyperedges, that is, $(i_1, i_2, i_3) \in E$ if there exists a hyperedge among vertices $i_1, i_2, i_3$.

In hyper-LSM, each vertex $i \in V$ is associated with an unknown latent position vector $u_i^* \in \mathbb{R}^r$. Similarly, the probability of generating hyperedge $(i_1, i_2, i_3)$ only depends solely on the latent positions. Suppose $U^* = (u_1^*, \ldots, u_n^*)^\top$ satisfying $n^{-1}U^\top U^* = I_r$ for identifiability. Let $A \in \{0, 1\}^{n \times n \times r}$ be the adjacency tensor of $G$. We assume there exists an unknown interaction $r \times r \times r$ tensor $C^*$ and a known link function such that

$$[A]_{i_1,i_2,i_3}^{\text{ind}} \overset{\text{Bernoulli}}{\sim} \text{Bernoulli}\left(\left\{ g(C^* \cdot \left[ U^*, U^*, U^* \right]) \right\}_{l \in [L]} \right),$$

implying that $\mathbb{E}A = g(\Theta^*)$ where $\Theta^* = C^* \cdot \left[ U^*, U^*, U^* \right]$. Therefore, the hyper-LSM is an hLSM with parameters $C^*, U^*$ and the link function $g(\cdot)$.

If $g(x) = x$ and $U^* \in \{0, 1\}^{n \times r}$ is a membership matrix such that $U^*_{ij} = 1$, the hyper-LSM reduces to the hypergraph stochastic block model (Ghoshdastidar and Dukkipati 2017; Kim, Bandeira, and Goemans 2018; Chien, Lin, and Wang 2018; Yuan et al. 2018). Moreover, if $U^*$ is the product of a diagonal matrix and a membership matrix, the hyper-LSM becomes the hypergraph degree corrected block model (Ke, Shi, and Xia 2019).

### 2.3. Dynamic Latent Space Model

A dynamic network is a times sequence of networks on the same set of vertices. There have emerged several approaches to model the temporal transition of network structures in dynamic
networks (Sarkar and Moore 2005; Xu 2015; Sewell and Chen 2015; Matias and Miele 2017). For simplicity, we only consider a simple dynamic network model which was often studied for change point detection in dynamic networks (Bhattacharjee, Banerjee, and Michailidis 2018).

Let a dynamic network $G = \{G_t\}_{t=1}^T$ compose of a sequence of $T$ networks on the same set of $n$ vertices $V$, where the binary graph $G_t := (V; E_t)$ represents the interaction at time $t$. Denote $A \in \{0, 1\}^{n \times n \times T}$ the adjacency tensor of $G$ whose $r$th slice $[A]_{i,:,:}$ is the adjacency matrix of $G_t$. For simplicity, we assume the network structures only change at $m \ll T$ unknown time points $[t_j]_{j=1}^m$, called change points (Bhattacharjee, Banerjee, and Michailidis 2018; Wang, Yu, and Rinaldo 2018). Here, $t_1 = 1$ and hence the initial network is always identified as a change point. The main task is to identify the other $m - 1$ change points and also recover underlying network structures, for example, the latent positions. For each $j \in [m]$ and $t \in [t_j, t_{j+1})$, we assume $G_t$ is generated from the same latent space model with the latent positions $U_j \in \mathbb{R}^{n \times q_j}$ and interaction matrix $C_j \in \mathbb{R}^{q_j \times q_j}$. We assume $n^{-1}U_j^\top U_j = I_q$ for identifiability and the network at each time point is independently sampled from the others. Denote $\hat{U} = \{U_1^\top, \ldots, U_m^\top\} \in \mathbb{R}^{n \times q}$ with $\hat{q} = \sum_{j=1}^m q_j$, whose rows reflect the global similarity between vertices throughout all the time. One can similarly define the interaction tensor as MMLSM of Section 2.1. Consequently, this simple dynamic LSM can be viewed as a special case of MMLSM in that the network layers between two consecutive change points are sampled from an identical latent space model.

3. Maximum Likelihood Estimation by Tensor Decomposition

In hLSM, with the observed adjacency tensor generated by model (4), our goal is to estimate the latent positions. In view of the low-rank structure of $\Theta^*$, a natural solution is the maximum likelihood estimator (MLE) with low-rank constraint. Let $\ell_n(\cdot) : \mathbb{R}^{n_1 \times n_2 \times n_3} \rightarrow \mathbb{R}$ be the negative log-likelihood for the distribution in hLSM, depending on the choice of a link function $g(\cdot)$. Given the observed binary adjacency tensor $A$ and a choice of latent parameters $\Theta \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, the corresponding negative log-likelihood is

$$
\ell_n(\Theta) = - \sum_{i,j \in [n_1], j \in [3]} (|A|_{i,j,3} \log g((\Theta)_{i,j,3})) + (1 - |A|_{i,j,3}) \log (1 - g((\Theta)_{i,j,3}))).
$$

Under mild regularity conditions on $g(\cdot)$, for example, strictly increasing monotonicity, the loss function $\ell_n(\cdot)$ is convex in $\Theta$. More details can be found in Section 4. Based on (7), we formulate the rank-constrained as follows

$$
\min_{\Theta} \ell_n(\Theta) \quad \text{subject to } \text{rank}(\Theta) \leq (r_1, r_2, r_3),
$$

where $\text{rank}(\cdot)$ denotes the Tucker ranks of a tensor. While the (unconstrained) objective function in problem (8) is usually convex, the rank-constrained feasible set is nonconvex. This rank constraint implies the existence of a low-rank decomposition $\Theta = C \cdot [U, V, W]$ with $U \in \mathbb{R}^{n_1 \times r_1}, V \in \mathbb{R}^{n_2 \times r_2}$ and $W \in \mathbb{R}^{n_3 \times r_3}$. Thus, the problem (8) is essentially boiled down to a generalized low-rank tensor decomposition which has been intensively investigated in the literature, for example, the penalized jointly gradient descent in Han, Willett, and Zhang (2020), the Riemannian gradient descent in Cai, Li, and Xia (2021), the alternating minimization in Wang and Li (2020) and Hu, Lee, and Wang (2019) and so on. These prior works all take advantage of the specific forms of decomposition of $\Theta$.

We propose a local algorithm for solving problem (8) by projected gradient descent on Grassmannian. The Grassmanian $Gr(n, r)$ is the collection of all $r$-dimensional subspaces in $\mathbb{R}^n$. The Stiefel manifold $St(n, r) = \{U : U^\top U = I_r\}$ is the set of orthonormal $r$-frames in $\mathbb{R}^n$. $Gr(n, r)$ can be obtained by identifying those matrices in $St(n, r)$ whose columns span the same subspace (a quotient manifold), (Edelman, Arias, and Smith 1998). Note that any $U \in Gr(n, r)$ satisfies that $U^\top U$ is identity. Thus, $Gr(n, r)$ naturally serves as the feasible set for the latent positions in hLSM (4) where $n^{-1/2}U^* \in Gr(n_1, r_1)$. Sometimes $U$ has a bounded incoherence constant so that its row-wise norm is small. To this end, for any $\delta \in (0, 1)$, we denote $Gr(n, r, \delta)$ the set of $U \in Gr(n, r)$ such that $\|U\|_{2,\infty} \leq \delta$. Equipped with Grassmannians and by taking advantage of the incoherence property, we reformulate the problem (8) as

$$
\min_{C, U, V, W \in \Theta} \ell_n(C \cdot [U, V, W])
$$

subject to $U \in Gr(n_1, r_1, \delta_1), V \in Gr(n_2, r_2, \delta_2), W \in Gr(n_3, r_3, \delta_3),$

where $\delta_i \in (0, 1)$ are tuning parameters. We show in Section 4 that, under mild conditions and given fixed $U, V, W$, the objective function of (9) is convex with respect to $C$. Since $C$ is low-dimensional, optimizing $C$ is computationally efficient. Thus, the major computation challenge lies in the search for optimal $U, V,$ and $W$.

The problem (9) is still highly nonconvex and solvable only locally where the gradient descent algorithm is often favored. Unfortunately, a naive gradient descent algorithm cannot ensure that the iterated estimations still (a) remain on Grassmannian; and (b) comply with the incoherence condition. The first issue can be resolved by considering the geodesic gradient descent on Grassmannian (Edelman, Arias, and Smith 1998; Xia and Yuan 2019; Keshavan, Montanari, and Oh 2010), but this approach is typically burdensome in computation and greatly complicates theoretical analysis. The second issue is simpler to resolve, for instance, by penalization (Xia and Yuan 2019) or projection (Ke, Shi, and Xia 2019; Han, Willett, and Zhang 2020).

We now propose our approach, based on the projected gradient descent on Grassmannians, for locally optimizing the problem (9). Our algorithm consists of three steps in every iteration.

- **Step 1.** At $t$th iteration, given the current estimate $\Theta(t) = C(t)$, $U(t) = [U^t, V^t, W^t]$, we calculate the gradients $\nabla_U \ell_n(\Theta(t)), \nabla_V \ell_n(\Theta(t))$ and $\nabla_W \ell_n(\Theta(t))$. With a properly chosen step-size $\eta > 0$, we update the estimate by gradient descent and obtain $\tilde{U}^t$ by the left singular vectors of $U^{t-1} - \eta \nabla_U \ell_n(\Theta(t))$. This is equivalent to projecting $U^{t-1} - \eta \nabla_U \ell_n(\Theta(t))$ onto the Grassmannian and thus $\tilde{U}^t \in Gr(n_1, r_1)$.

- **Step 2.** The updated $\tilde{U}^t$ from Step 1 may have a large incoherence coefficient. To reinstate incoherence, we impose a regularization that rescales all row $\ell_2$-norms higher than...
δ. Formally, for any $U \in \text{Gr}(n, r)$, define the regularization operator by
\[
\text{Reg}_\delta(U) := D_U U, \quad \text{where}
\]
\[
D_U = \text{diag} \left( \min \left\{ \delta, \|U_{1:} \| \right\}, \ldots, \min \left\{ \delta, \|U_n: \| \right\} \right)
\]
By definition, the output satisfies $\|\text{Reg}_\delta(U)\|_{2,\infty} \leq \delta$. Then we set $U^{(t+1)}$ to be the left singular vectors of $\text{Reg}_\delta(U^{(t)})$, which provably satisfies $U^{(t+1)} \in \text{Gr}(n_1, r_1, 2\delta)$. Update $V^{(t+1)}$ and $W^{(t+1)}$ using the same procedure described above.

**Step 3.** With the updated $U^{(t+1)}, V^{(t+1)}$, and $W^{(t+1)}$, we find the core tensor $C^{(t+1)}$ by solving $\arg \min_{C: \|C\|_{F,\ell}} \ell_n(C)$, where $\xi > 0$ is a tuning parameter.

Recall that $C \in \mathbb{R}^{n \times n^2 \times n}$ is low-dimensional and the objective function is convex (see more details in Section 4) in $C$, the update $C^{(t+1)}$ can be efficiently found by Newton-Raphson algorithm.

The implementation details of our algorithm are summarized in Algorithm 1. We note that the gradient can be explicitly computed by $\nabla U \ell_n(\Theta) = \mathcal{M}_1(\nabla \ell(\Theta))(W \otimes V) \mathcal{M}_1(C)$, where recall that $\Theta = C \cdot [U, V, W]$.

**Algorithm 1** Projected Gradient Descent on Grassmannians

**Input:** Tuning parameters $\delta_1, \delta_2, \delta_3, \xi > 0$; learning rate $\eta > 0$; maximum iterations $t_{\text{max}}$; initialization $U^{(0)} \in \text{Gr}(n_1, r_1, \delta_1)$, $V^{(0)} \in \text{Gr}(n_2, r_2, \delta_2)$, $W^{(0)} \in \text{Gr}(n_3, r_3, \delta_3)$; $C^{(0)} \leftarrow \arg \min_{C: \|C\|_{\ell} \leq \xi} \ell_n(C \cdot [U^{(0)}, V^{(0)}, W^{(0)}])$

**Output:** $\Theta, U, V, W$

for $t = 1, 2, \ldots, t_{\text{max}}$ do

1. $\Theta^{(t)} \leftarrow C^{(t-1)}$, $U^{(t-1)}, V^{(t-1)}, W^{(t-1)}$

2. Gradient descent:

   $\tilde{U}^{(t-1)} \leftarrow \text{SVD}(U^{(t-1)} - \nabla_U \ell_n(\Theta^{(t-1)}))$

   $\tilde{V}^{(t-1)} \leftarrow \text{SVD}(V^{(t-1)} - \nabla_V \ell_n(\Theta^{(t-1)}))$

   $\tilde{W}^{(t-1)} \leftarrow \text{SVD}(W^{(t-1)} - \nabla_W \ell_n(\Theta^{(t-1)}))$

3. Regularization:

   $U^{(t)} \leftarrow \text{SVD}(\text{Reg}_\delta(\tilde{U}^{(t-1)}))$

   $V^{(t)} \leftarrow \text{SVD}(\text{Reg}_\delta(\tilde{V}^{(t-1)}))$

   $W^{(t)} \leftarrow \text{SVD}(\text{Reg}_\delta(\tilde{W}^{(t-1)}))$

4. Compute $C^{(t)} \leftarrow \arg \min_{C: \|C\|_{\ell} \leq \xi} \ell_n(C \cdot [U^{(t)}, V^{(t)}, W^{(t)}])$

end for

Set $\tilde{C} \leftarrow C^{(t)}$, $\tilde{U} \leftarrow U^{(t)}$, $\tilde{V} \leftarrow V^{(t)}$, $\tilde{W} \leftarrow W^{(t)}$, $\tilde{\Theta} \leftarrow \tilde{C} \cdot [\tilde{U}, \tilde{V}, \tilde{W}]$

4. **Convergence and Estimation Accuracy**

We now present the convergence performances of Algorithm 1 and the general statistical error bounds of the final estimates of latent positions. Then we apply these results to several specific hlSM models and elucidate the accuracy of estimated latent positions.

### 4.1. Regularity Conditions on the Link and Loss Functions

The link function $g(\cdot)$ plays a decisive role in the convergence of Algorithm 1. It determines the geometry of loss function $\ell_n(\cdot)$. Define
\[
g_+(x) := \left(\frac{g'(x)}{g(x)}\right)^2 - \frac{g''(x)}{g(x)} \quad \text{and} \quad g_-(x) := \left(\frac{g'(x)}{1-g(x)}\right)^2 + \frac{g''(x)}{1-g(x)},
\]
which are the second-order derivative of $-\log g(x)$ and $-\log(1-g(x))$.

**Assumption 1.** Assume that for any small $\alpha > 0$, there exist $\gamma_\alpha, \beta_\alpha > 0$ depending only on $\alpha$ and $g(\cdot)$ such that
\[
\min \left\{ \inf_{|x| \leq \alpha} g_+(x), \inf_{|x| \leq \alpha} g_-(x) \right\} \geq \gamma_\alpha \quad \text{and} \quad \max \left\{ \sup_{|x| \leq \alpha} g_+(x), \sup_{|x| \leq \alpha} g_-(x) \right\} \leq \beta_\alpha
\]
The quantities $\gamma_\alpha$ and $\beta_\alpha$ are often sensitive to $\alpha$. For instance, if $g(x) = (1 + e^{-x/\alpha})^{-1}$ is the logistic link with a global scaling $\sigma > 0$, we have $\gamma_\alpha = e^{\alpha/\sigma}[\sigma(1 + e^{\alpha/\sigma})]^{-3}$ and $\beta_\alpha = 1/(4\sigma^2)$; if $g(x) = \Phi(x)$ is the probit link, we have $\gamma_\alpha \approx (\alpha + 0.1)(2\pi)^{-1/2} \cdot \alpha^{-2}$ and $\beta_\alpha \approx 0.6$. These examples suggest that $\beta_\alpha^{-1}$ increases faster as $\alpha$ becomes larger.

We now state our main assumption on the latent positions $U^*, V^*, W^*$ and the underlying low-rank tensor $\Theta^* \in [U^*, V^*, W^*]$ of hlSM (4).

**Assumption 2.** Assume that $U^*, V^*, W^*$ are incoherent with constants upper bounded by $\mu_0 > 0$, that is, $\text{Incoh}(U^*), \text{Incoh}(V^*), \text{Incoh}(W^*) \leq \mu_0$. Also, the largest singular value of $C^*$ is upper bounded by $\tilde{\lambda}(C^*) \leq \mu_0^{-2}(r_1r_2r_3)^{-1/2}$.

In hlSM, Assumption 2 implies that $\|U^*\|_{2,\infty} \leq \mu_0^{1/2}$, $\|V^*\|_{2,\infty} \leq \mu_0^{1/2}$ and $\|W^*\|_{2,\infty} \leq \mu_0^{1/2}$. Together with the upper bound of $\tilde{\lambda}(C^*)$, Assumption 2 implies that $\|\Theta^*\|_{\infty} \leq \alpha$.

Then, Assumption 1 implies that entry-wisely, $\gamma_\alpha \leq g_+(\Theta^*) \leq \beta_\alpha$ and $\gamma_\alpha \leq g_-(\Theta^*) \leq \beta_\alpha$. This is crucial to ensure the strongly convexity and smoothness of the loss function around the truth. The following lemma is straightforwardly implied by Assumption 1, and thus we omit its proof.

**Lemma 1.** Under Assumption 1, the loss function $\ell_n(\cdot)$ is $\gamma_\alpha$-strongly convex and $\beta_\alpha$-smooth on the set $K_\alpha := \{\Theta \in \mathbb{R}^{n_1 \times n_2 \times n_3} : \|\Theta\|_{\infty} \leq \alpha\}$, that is,
\[
\|\nabla \ell_n(\Theta_1) - \nabla \ell_n(\Theta_2), \Theta_1 - \Theta_2 \|_F \geq \gamma_\alpha \|\Theta_1 - \Theta_2\|_F^2
\]
\[
\|\nabla \ell_n(\Theta_1) - \nabla \ell_n(\Theta_2)\|_F \leq \beta_\alpha \|\Theta_1 - \Theta_2\|_F
\]
for any $\Theta_1, \Theta_2 \in K_\alpha$.

The next lemma investigates the update of $C$ in the main iteration of Algorithm 1 and quantifies the convexity of the objective function in C, given properly updated $U, V, W$. We relegate its proof to the Appendix, supplementary materials.
Lemma 2. Suppose Assumption 1 holds. Let \( U \in \text{Gr}(n_1, r_1, \delta_1), V \in \text{Gr}(n_2, r_2, \delta_2), W \in \text{Gr}(n_3, r_3, \delta_3) \) be fixed with \( \delta_j \leq \mu_0 (r_j/n_j)^{1/2} \). If we view \( \ell_{\alpha}(C) = \|U, V, W\| \) as a function of \( C \), then it is \( \gamma_{\alpha} \)-strongly convex on the set \( \{ C \in R_{\mathbb{H}1} \times R_{\mathbb{H}2} \times R_{\mathbb{H}3} : \| C \|_F \leq \alpha \mu_0^{-3} (n_1 n_2 n_3 r_1^{-1} r_2^{-1} r_3^{-1})^{1/2} \} \).

By Lemma 2, with a properly chosen tuning parameter \( \xi \), the objective function in the optimization program for updating \( C \) is strongly convex.

4.2. Error Bounds of Latent Position Estimates under General hLSM

Let \( (U^{(t)})_{t=1}^{\text{max}}, (V^{(t)})_{t=1}^{\text{max}}, (W^{(t)})_{t=1}^{\text{max}} \) be the iterative updates by Algorithm 1. Notice that, due to the column-normalization of SVD, \( U^{(t)} \) (and similarly, \( V^{(t)} \) and \( W^{(t)} \)) estimates \( n_1^{-1/2} U^* \) rather than \( U^* \), up to an unknown right-rotation. Therefore, we measure the error of \( U^{(t)} \) by the chordal-Brownian norm distance. Formally, define \( d_t(U^{(t)}, n_1^{-1/2} U^*) := \min_{O \in O_r} \| U^{(t)} - n_1^{-1/2} U^* O \|_F \), where \( O_r \) is the set of \( r \times r \) orthogonal matrices. Define the error measurements for \( V^{(t)} \) and \( W^{(t)} \) similarly. Then we define the overall error \( D_t \) at the \( t \)th iteration to be

\[
D_t^2 := d_t^2(U^{(t)}, n_1^{-1/2} U^*) + d_t^2(V^{(t)}, n_2^{-1/2} V^*) + d_t^2(W^{(t)}, n_3^{-1/2} W^*)
\]

The statistical error of the final estimate \( (\hat{U}, \hat{V}, \hat{W}) \) depends on the gradient of the loss function at the truth \( \Theta^* \). Let \( \mathbf{r} = (r_1, r_2, r_3) \) denote the Tucker ranks of \( \Theta^* \). The stochastic error of the final estimate in hLSM is determined by \( Err_t := \sup_{\| \mathbf{x} \|_F \leq 1, \text{rank}(\mathbf{x}) = r} \| \nabla \ell_{\alpha}(\Theta^*) - \nabla \ell_{\alpha}(\Theta^*) \|_F \), where the expectation is taken w.r.t. the randomness of \( A \). Under hLSM, we have \( \mathbb{E} \nabla \ell_{\alpha}(\Theta^*) = 0 \) since

\[
[\nabla \ell_{\alpha}(\Theta^*)]_{i_1 i_2 i_3} = \frac{g([\Theta^*]_{i_1 i_2 i_3}) - [A]_{i_1 i_2 i_3}}{g(\Theta^*)} \cdot [\Theta^*]_{i_1 i_2 i_3}.
\]

Recall that \( \| \Theta^* \|_\infty \leq \alpha \) under Assumption 2. Define \( \zeta_\alpha := \sup_{x \in \mathbb{R}} \left[ \frac{g(\alpha x)}{g(x)} (1 - g(x)) \right]^{-1} \). Then, the entries of \( \nabla \ell_{\alpha}(\Theta^*) \) are independent centered sub-Gaussian random variables which are uniformly upper bounded by \( \zeta_\alpha \). We remark that the quantity \( \zeta_\alpha \) is a nondecreasing function of \( \alpha \). In particular, \( \zeta_\alpha = 1 \) for logistic link \( g(x) = (1 + e^{-x})^{-1} \) and \( \zeta_\alpha \leq 2 \alpha^{-1} (\alpha/\alpha + 1) \) for probit link \( g(x) = \Phi(x) \).

The following lemma characterizes the magnitude of the stochastic error \( Err_t \), whose proof is deferred to the Appendix, supplementary materials.

Lemma 3. Under Assumption 2, there exist an absolute constant \( \zeta_\alpha > 0 \) such that with probability at least \( 1 - \exp(-c_0 (r_1 r_2 r_3 + n_1 r_1 + n_2 r_2 + n_3 r_3)) \),

\[
Err_t \leq \zeta_\alpha \left( \frac{r_1 r_2 r_3 + \sum_{k=1}^3 n_k r_k}{\sqrt{1}} \right)^{1/2}
\]

Denote \( \kappa_0 := \sqrt{\Lambda(C^*)/\Lambda(C^*)} \) the condition number of \( C^* \) and \( \bar{r} = \max_{1 \leq j \leq 3} r_j \). The following theorem shows that, with good initializations and appropriately chosen tuning parameters, Algorithm 1 converges linearly and the error of final outputs only depends on the signal strength \( \Lambda(C^*) \) and the stochastic error \( Err_t \).

Theorem 1. Suppose Assumption 1–2 hold in hLSM (4). Assume that there exist constants \( c_4 \in (0, 1) \) and \( C_{\alpha, 4} > 0 \) depending only on \( \alpha \), and absolute constants \( c_1, c_2 \in (0, 1) \) with \( c_1 < c_2 \), the following conditions hold:

1. Initialization error:
\[
D_0^2 \leq c_4 \kappa_0^2 \zeta_\alpha^2 \bar{r}^{-3}
\]

2. Signal-to-noise ratio:
\[
(n_1 n_2 n_3)^{1/2} \cdot \frac{\Lambda(C^*)}{\text{Err}_t} \geq c_1^{-1} C_{\alpha, 4} \kappa_0^2 \bar{r}^{-3/4}
\]

Then \( \zeta_\alpha = O(1) \) and \( n_1 \geq n_2 \geq n_3 \), it implies that \( D_{\text{max}} \) converges to zero as long as \( n_1 n_2 n_3 \Lambda(C^*)/\text{Err}_t \rightarrow \infty \). Put differently, the estimation error of the latent positions diminishes very quickly as the network size grows, which also matches our observations in simulation studies.

4.3. Error Bound of Latent Position Estimates for Specific hLSMs

We now apply Theorem 1 to the specific examples of hLSM. Here and after, for notational simplicity, we assume the maximum number of iterations \( t_{\text{max}} = O(\log(n_1 n_2 n_3 \Lambda(C^*)/\text{Err}_t)) \) under the general LSM model (4). Throughout this section, we assume the initialization condition of Theorem 1 holds.

4.3.1. Application 1: Mixture Multilayer Latent Space Model (MMLSM)

Let \( \hat{U} = (n_1^{-1/2} \hat{U}^*) \Sigma_{\hat{U}} R_{\hat{U}}^T \) denote the thin SVD of \( \hat{U} \), where the \( r \times r \) diagonal matrix \( \Sigma_{\hat{U}} \) contains the singular values of \( \hat{U} \). To characterize the signal strength of \( C^* \), define \( \hat{C} = C \times_1 R_{\hat{U}}^T \times_2 R_{\hat{U}}^T \).

As a result, \( C^* \) can written in terms of \( \hat{C} \) as follows:

\[
C^* = (nL_{\hat{U}}^{-1}) \cdot \left( \hat{C} \times_1 \Sigma_{\hat{U}} \times_2 \Sigma_{\hat{U}} \times_3 \text{diag}(L_{\hat{U}}^{1/2}, \ldots, L_{\hat{U}}^{1/2}) \right).
\]
It turns out that \( \Delta(C^*) \geq n^{-1}L^{-1/2}2\sigma^2_{\min} \left( U^T \Delta(\hat{C}) \right) \min_{1 \leq j \leq m} L_j^{1/2} \).

Denote \( \kappa_U \) the condition number of \( \hat{U} \).

The following corollary is an immediate conclusion from combining Theorem 1 and Lemma 3.

**Corollary 1** (Error bounds of estimating latent positions in MMLSM). Suppose Assumption 1–2 hold. Let \( \hat{U} \) be the output of Algorithm 1. Denote the signal strength of \( \hat{C} \) by \( c_\alpha = \Delta(\hat{C}) \). If the network cluster sizes are balanced \( \min_{1 \leq j \leq m} L_j \propto \frac{L}{m} \), then there exists an absolute constant \( c_0 > 0 \) such that with probability at least \( 1 - \exp \left( -c_0 (2nr + L + m + nr^2) \right) \),

\[
d^2(\hat{U}, n^{-1/2}\hat{U}^*) \leq C_\alpha c_\alpha^2 \kappa_U^{-1}(r \vee m)^{1/2} \left( \frac{2r}{m} n + L + r^2 \right) c_\alpha^2
\]

provided that

\[
n^2L \geq C_\alpha c_\alpha^2 \kappa_U^{-1}(r \vee m)^{1/2} \left( \frac{2r}{m} n + L + r^2 \right) c_\alpha^{-2}
\]

where \( C_\alpha > 0 \) is a constant depending only on \( \alpha \).

To gain more insight, let us consider a simple setting where \( r, m, c_\alpha, \kappa_U = O(1) \). The error rate in (10) simplifies to \( (n + L)(c_\alpha^2 n^2 L) \), and we observe an interesting phase transition: (a) when the number of layers \( L \) is small compared to \( n \), that is \( L = O(n) \), the error rate is dominated by the first term \( 1/(c_\alpha^2 nL) \). In this phase, increasing the number of nodes or the number of layers can both improve the estimation of latent positions. (b) when \( L \gg n \), the error rate would be bottlenecked by the second term \( 1/(c_\alpha^2 n^2) \), which does not depend on \( L \) any more. Consequently, increasing the number of layers can no longer improve the estimate of latent positions. This phase transition is also empirically confirmed by our simulation studies, see Section 5. The latter phase seems unexpected since it implies that, beyond certain threshold, increasing the number of layers brings diminishing benefits to the estimation of latent positions. This result, actually, is an outcome due to both the difficulty of the mixture model and the limitation of tensor methods.

The mixture nature of MMLSM underlines the importance of estimating the \( L \times m \) matrix \( W^* \). However, our tensor method jointly estimates \( \hat{U}^* \) and \( W^* \), and the errors of \( \hat{U} \) and \( \hat{W} \) are thus intertwined. Clearly, when \( L \gg n \), estimating \( W^* \) is more difficult than estimating \( \hat{U}^* \). Therefore, in the latter phase, the error rate reflects the difficulty of recovering \( W^* \) rather than estimating \( \hat{U}^* \). This phenomenon can be easily understood from Theorem 1 under general hLSMs. Indeed, one can expect that for a more general tensor \( A \in \mathbb{R}^{n_1 \times n_2 \times n_3} \), this error bound would become \( (n_1 + n_2 + n_3)/(n_1 n_2 n_3) \). Without loss of generality if \( n_1 \gg n_2, n_3 \), then the dominating term would be \( 1/(n_2 n_3) \) and increasing \( n_1 \) would only bring diminishing benefits.

**Comparison with other multi-layer network models.** In Paul and Chen (2020b), the authors proposed a multi-layer stochastic blockmodel (MLSBM) with a consensus community membership matrix \( Z \) across all layers. Under the dense graph regime, they showed that \( d^2 \left( \hat{U}^*, \hat{U}^*_0 \right) = O(n^{-1/2} + L^{-1/2}) \) with high probability where \( \hat{U}^*_0 = \mathbb{E}(Z^T Z)^{-1/2} \). The rate is sub-optimal compared to our Corollary 1. Arroyo et al. (2021) introduced the common subspace independent-edge (COSIE) random graphs model, and derived \( \mathbb{E}(d^2(\hat{V}, V)^2) = O((nL)^{-1} + n^{-2}) \) under multilayer SBM with common community assignments (represented by the common subspace \( V \)) as a special example. But their theoretical result for the general case cannot be simply stated in terms of \( n \) and \( L \). A more comparable model to ours shall be the latent space model with a common latent position \( U^* \) introduced in Zhang, Xue, and Zhu (2020). The convergence rate of their proposed estimator, which reads \( d^2 \left( \hat{U}, U^* \right) = O((nL)^{-1} + n^{-2}) \) and holds with high probability, is derived only under the regime \( L = O(n) \), and it is unclear if their analysis can be extended to the case \( L \gg n \).

We can also recover the network classes \( S \) by applying standard K-means clustering to the rows of \( \hat{W} \) from Algorithm 1. Given an \( \hat{S} = \{ \hat{S}_i \}_{i=1}^L \), the estimator of \( S \), we use the average Hamming distance to measure its accuracy:

\[
\mathcal{L}(\hat{S}, S) = \min_{\tau: a \text{ permutation}} \frac{1}{L} \sum_{l=1}^L \sum_{i=1}^m \mathbb{1} \{ \hat{S}_{\tau(i)} \neq S_i \}
\]

**Theorem 2** (Error bounds of network clustering in MMLSM). Under the conditions of Corollary 1, there exists a global constant \( c_0 > 0 \) such that with probability at least \( 1 - \exp \left( -c_0 (2nr + L + m + nr^2) \right) \),

\[
\mathcal{L}(\hat{S}, S) \leq C_\alpha c_\alpha^2 \kappa_U^{-1}(r \vee m)^{1/2} \left( \frac{2r}{m} n + L + r^2 \right) c_\alpha^{-2}
\]

provided that

\[
n^2L \geq C_\alpha c_\alpha^2 \kappa_U^{-1}(r \vee m)^{1/2} \left( \frac{2r}{m} n + L + r^2 \right) c_\alpha^{-2}
\]

where \( C_\alpha > 0 \) is a constant depending only on \( \alpha \).

Corollary 1 and Theorem 2 suggest that, under similar mild signal strength conditions, both the global latent positions and the layer labels can be consistently recovered. Here we have the similar understanding as in Corollary 2 that the accuracy is bottlenecked by the asymptotically smaller one between \( n \) and \( L \).

**Remark 1.** After obtaining the layer labels, one can further estimate the local latent positions for each LSM(\( U_j, C_j \)). Since the layers with equal labels are assumed to be sampled from the same LSM, it is unnecessary to apply tensor methods (the factor corresponding to the third dimension becomes trivially constant). Interested readers may refer to Zhang and Cao (2017), Zhang, Xue, and Zhu (2020) and references therein for more details.

### 4.3.2. Application 2: Hypergraph Latent Space Model (Hyper-LSM)

Similar to Corollary 1, we have the following result, whose proof is straightforward.

**Corollary 2** (Error bounds of estimating latent position in hyper-LSM). Suppose Assumption 1–2 hold. Let \( \hat{U} \) be the output of Algorithm 1. Denote the signal strength of \( C^* \) by \( c_\alpha = \Delta(C^*) \).
Then there exists some absolute constant \( c_0 > 0 \) such that with probability at least 1 − \exp (−c_0 (k^3 + 3nr)),

\[
d_t(U, n^{-1/2}U^\ast) \leq C_d \zeta_n r^{1/2}(3nr + r^2) n^2 \zeta_n^2
\]

provided that \( n^2 \geq C_c \zeta_n r^{3/2} (3 + r^2 n^{-1}) \zeta_n^{-2} \) with the constant \( C_c > 0 \) depending only on \( \alpha \).

If \( \zeta_n, r = O(1) \), the error rate \( (11) \) simplifies to \( 1/(n^2 \zeta_n^2) \), where we recall that in an hyper-LSM, by definition \( L = n \). This bound diminishes quadratically in \( n \). Similarly, the minimal signal strength requirement \( c_s \) also decreases linearly with respect to \( n \). Most aforementioned literature concern the task of community detection (Ghoshdastidar and Dukkipati 2017; Kim, Bandeira, and Goemans 2018; Chien, Lin, and Wang 2018; Yuan et al. 2018) and thus are not directly comparable to our results. Nonetheless, Ke, Shi, and Xia (2019) revealed that the factor matrix \( \mathbf{X} \) obtained by Tensor-SCORE achieves the convergence rate of \( d_t(U, \mathbf{X}) = O((\log(n) n^{-3} n^{-2}) \zeta_n^{-2}) \) with high probability, where \( \zeta_n \) is an analogy to \( c_s \) with a slightly different definition.

### 4.3.3. Application 3: Dynamic Latent Space Model (Dynamic LSM)

Lastly, we consider the change point detection in dynamic latent space model. With the output \( \mathbf{W} \) of Algorithm 1, we perform a row-wise screening to identify the change points. More specifically, we iteratively compare the difference of two consecutive rows of \( \mathbf{W} \) in \( \ell_2 \) norm, and for all \( t \in [T] \), \( t + 1 \) is identified as a change point if and only if \( \| \mathbf{W}_{t-1} \|_2 \geq \varepsilon \) for some tuning parameter \( \varepsilon > 0 \). Define the \( r \times r \times m \) tensor \( \mathbf{C} \) in the same fashion as in MMLSM, and we can have the following result.

**Theorem 3** (Exact detection of change points in dynamic LSM). Suppose Assumption 1–2 hold. Denote the signal strength of \( \mathbf{C} \) by \( c_s = \Delta(c) \). If the time intervals between neighboring change points are balanced \( \min_{1 \leq j \leq m} T_j \approx T/m \), then there exist absolute constants \( c_0, c_1, c_2 > 0 \) such that by choosing \( \varepsilon = \max(c_1 (T/m)^{1/2}, c_2 (T/m)^{1/2}) \), change points can be exactly detected with probability at least \( 1 - \exp (-c_0 (2nr + Tm + mr^2)) \), provided that \( nT \geq C_c \zeta_n r^2 (r \vee m)^{1/4} (2nr + Tm + mr^2)^{1/2} \zeta_n^{-1} \) where \( C_c > 0 \) is constant depending only on \( \alpha \).

If \( k_0, r = O(1) \) and \( m = O(n) \), by Theorem 3, in order to exactly detect those change points, the minimal signal strength requirement becomes \( c_s n r^2 \geq C m^{1/2} \) for some absolute constant \( C > 0 \).

Our result characterizes the probability of exact change point recovery, which is a natural consequence of accurate latent position estimation, and is different from the noisy recovery error rate in Wang, Yu, and Rinaldo (2018). Therefore, the signal strength assumption of our Theorem 3 and the counterpart of Wang, Yu, and Rinaldo (2018) are not directly comparable. Zhao, Chen, and Lin (2019) proposed a screening and thresholding change point detection algorithm, whose signal strength \( \Delta^* \) is defined as the minimal difference of adjacent probability link matrices in \( \| \cdot \|_{2, \infty} \) norm. The signal strength condition that guarantees the exact recovery of change points is \( \Delta^* \geq (\log n)^{1/2} n^{1/2} / n^{1/2} \) for any \( \delta > 0 \) if the minimum segment length \( D^* = \min_{1 \leq j \leq m+1} (T_j - T_{j-1}) = O(1) \), which is independent of \( T \). In fact, our result provide richer information about changes in network evolution that are not limited to sudden changes. For instance, our method is capable of revealing a dynamic network that shows rapid but continuous changes during change periods rather than change points. This pattern is not covered by most change detection literature in network analysis.

### 4.4. Warm Initialization

The key to the success of Algorithm 1 is a warm initialization satisfying the condition in Theorem 1. In this section, we present such an algorithm adapted from Davenport et al. (2014), which allows us to first reduces the original problem to the convex optimization in 1-bit matrix completion and then apply rank 1-HOSVD\(^2\) to have a low-rank approximation. An additional regularization step is applied to guarantee the incoherence. The details are summarized in Algorithm 2, which is theoretically justified by the following lemma.

**Algorithm 2** Initialization for Algorithm 1

**Input:** Data tensor \( \mathbf{A} \), rank \( r = (r_1, r_2, r_3) \), regularization parameters \( \delta_1, \delta_2, \text{and } \delta_3 \).

**Output:** \( U(0), V(0), W(0) \)

1. Let \( k^* = \arg \max_{k=1,2,3} n_k \). Solve the following convex optimization problem:

\[
\hat{\Theta} = \arg \min_{\Theta} \ell_n(\Theta) \quad \text{subject to} \quad \| M_k^\ast(\Theta) \|_\infty \leq \alpha \sqrt{n_1 n_2 n_3 r_k} \quad \text{and} \quad \| \Theta \|_\infty \leq \alpha \quad (12)
\]

2. Apply rank 1-HOSVD on \( \hat{\Theta} \), get the leading \( (r_1, r_2, r_3) \) singular vectors \( \hat{U}, \hat{V}, \hat{W} \).

3. Regularization:

\[
\begin{align*}
\text{SVD(Reg}_{\delta_1}(\hat{U})); & \quad V(0) \leftarrow \text{SVD(Reg}_{\delta_2}(\hat{V})); \\
W(0) \leftarrow \text{SVD(Reg}_{\delta_3}(\hat{W}) & \end{align*}
\]

**Lemma 4.** Let \( \delta_k = 2 \mu_0(r_k/n_k)^{1/2} \) for \( k \in [3] \), then with probability at least \( 1 - C(n + n^2)^{-1} \) for some absolute constant \( C > 0 \), the output of Algorithm 2 satisfies the initialization condition in Theorem 1 and \( U(0), V(0), W(0) \) are \( 2 \mu_0 \)-incoherent, if the following signal-to-noise condition holds:

\[
(n_1 n_2 n_3)^{1/2} \frac{\Delta(c^\ast)}{\text{Err}_r} \geq C_{\alpha} \left( \frac{r_k n_1 n_2 n_3 (n + n^-)^{1/4}}{r_1 r_2 r_3 + \sum_{k=1}^3 n_k r_k} \right)^{1/2} \kappa_0^{-3/2} \quad (13)
\]

for some constant \( C_{\alpha} > 0 \) depending only on \( \alpha \) and the same constant \( c_1 \) in Theorem 1, where \( \hat{n} = \max_{k=1,2,3} n_k \) and \( n^- = n_1 n_2 n_3 / \hat{n} \).

Considering the case \( n_k \approx n \) and \( \kappa_0, r_k = O(1) \) for all \( k \), the SNR condition (13) simplifies to \( n^{3/2} \Delta(c^\ast) \geq n^{3/4} \text{Err}_r \).

This is much stronger than the condition necessary to ensure the convergence of Algorithm 1. In fact, Theorem 1 only requires

\[\text{2 More specifically, the rank 1-HOSVD on a } m \text{-way tensor } \mathbf{H} \text{ is conducted by extracting the top-} r \text{ left singular vectors of the matrices } M_j(\mathbf{H}) \text{ for } j \in [m], \text{ where } r = (r_1, \ldots, r_m).\]


\[ n^{3/2} A(C^*) \gtrsim O(1). \] Gaps of this kind are prevalent in tensor literature (Zhang and Xia 2018; Xia and Yuan 2019; Cai, Li, and Xia 2021). Investigating the optimal SNR condition is challenging under our general framework, and hence we spare no additional efforts in this regard.

5. Simulations on Synthetic Higher-order Networks

We showcase the performances of Algorithm 1 on synthetic higher-order networks. Due to space limit, we only present here the results on general higher-order LSM. Numerical simulations on multi-layer, hypergraph and dynamic networks are relegated to the supplementary file.

5.1. Simulation 1: General Higher-Order LSM’s

Without loss of generality, we only consider third-order networks for the general higher-order latent space model. The network sizes are fixed at \( n_k \equiv n = 50 \) unless specified and the dimension of latent space is fixed at \( r_k \equiv r = 3 \) for \( k = 1, 2, 3 \). We generate the low-rank parameter tensor \( \Theta^* \) as follows. We first generate a truncated standard normal tensor \( \tilde{\Theta} \in \mathbb{R}^{n \times n \times n} \) with \( (\tilde{\Theta})_{ijk} \overset{\text{iid}}{\sim} \text{TruncNorm}(0, 1; [-3, 3]), i, j, k \in [n], \) and then apply higher-order SVD to \( 10 \cdot \tilde{\Theta} \) with multilinear ranks \((r, r, r)\), which produces the core tensor \((n_1n_2n_3)^{1/3} C^* \in \mathbb{R}^{n \times n \times n} \) and factor matrices \( n_1^{-1/2} U^*, n_2^{-1/2} V^*, n_3^{-1/2} W^*. \) The parameter tensor is then set to \( \Theta^* = C^* \cdot [U^*, V^*, W^*]. \) The observed data tensor \( A \) has independent entries sampled from \( \text{Bernoulli}(g(\Theta^*/\sigma)) \) entry-wisely, where we set the link function \( g(\cdot) := \text{logit}(\cdot; \sigma) \) with a global scaling parameter \( \sigma \in \{0.1, 0.5, 1\}. \)

The computation of the projected gradient descent updates \( U^{(i)}, V^{(i)}, W^{(i)} \) is fast and memory-efficient. The main computation burden of Algorithm 1 comes from the update of the core tensor \( C^{(i)} \). Computing \( C^{(i)} \) can be recast as essentially estimating a generalized linear model, for example, logistic/probit regression with logit/probit link function. This step can be computationally demanding when \( n_1n_2n_3 \) is large. Fortunately, the number of parameters we desire to estimate is only \( r_1r_2r_3 \), comparatively much smaller than \( n_1n_2n_3 \). To alleviate the computation costs of this step and accelerate our algorithm, we accelerate by updating \( C^{(i)} \) using a small sub-sample input: \((|A|S_{i1}S_{i2}S_{i3}; |U^{(i)}|S_{i4}S_{i5}, |V^{(i)}|S_{i6}S_{i7}, |W^{(i)}|S_{i8}) \), where \( S_{ik} \subseteq [n_i], |S_{i}| \ll n_i \), instead of the original \((A; U^{(i)}, V^{(i)}, W^{(i)})\), except the last few iterations. This random sampling procedure allows to solve \( C^{(i)} \) via a much smaller scale logistic regression. We regard this method as an accelerated version of Algorithm 1. This accelerated Algorithm 1 can greatly improve speed at little cost of estimation accuracy—Figures 1 and 2 demonstrate that it enjoys almost same convergence and accuracy as the original algorithm. In these simulations, the sampling proportion is 0.1 and the algorithm runs 5 times faster than the original algorithm.

Figures 1 and 2 present the simulation results on convergence of our algorithm with iterations going on. Both figures report that our algorithm converges in around 20 iterations in terms of the error of latent positions estimates. In Figure 2, a smaller \( \sigma \) corresponds to the easier dense network setting, and our algorithm converges even faster. Moreover, the linear pattern at the early stages echos the linear convergence of Algorithm 1 predicted by Theorem 1.

Sparse network. Note that network sparsity is not studied in our framework. Here we present numerical simulations to show the impact of network sparsity. Figure 3 shows the performance of our algorithm on varying network sparsity. The error of latent positions and that of parameter tensor are both stable when network sparsity \( > 0.1 \). When the network is extremely sparse (especially when sparsity \( < 0.01 \)), the error of latent positions starts to increase rapidly and become unstable, whereas the relative error of \( \Theta \) remains acceptable. This gives evidence on the utility of our algorithm for sparse networks in estimating \( \Theta \).

6. Data Examples

In this section, we demonstrate the merits of our methods in node embedding and link predictions on two real-world datasets. For both applications, the logistic link function is used and High-Order Orthogonal Iteration (HOOI) (Zhang and Xia 2018; Ke, Shi, and Xia 2019) is used for initialization, whose results differ not much from that initialized by Algorithm 2.

6.1. Trade Flow Multi-Layer Network from UN Comtrade

The multi-layer network data are constructed based on the international commodity trade data collected from the UN Comtrade Database (https://comtrade.un.org). The dataset contains annual trade information for countries/regions from different continents in 2019, where, for ease of presentation, we only focus on the top representative 48 countries/regions ranked by the exports of goods and services in U.S. dollars. Each layer represents a different type of commodities classified into 97 categories based on the 2-digit HS code (https://www.foreign-trade.com/reference/hscode.htm). For every two nodes \( i \) and \( j \), we convert the weighted edge \( w_{i \rightarrow j} \geq 0 \) in the original data into one binary directed edge, that is, \( A_{i \rightarrow j} = 1(w_{i \rightarrow j} > 0) \). The adjacency tensor \( A \) is defined in the way such that \([A]_{ij} = 1 \) if country \( i \) exports to country \( j \) in terms of commodity type \( l \). We remove empty layers and obtain a binary adjacency tensor \( A \) of size \( 48 \times 48 \times 97 \).

We apply our Algorithm 1 to \( A \) and obtain an estimated \( \hat{W} \). Empirical evidence (the numerical scales of the leading eigenvalues and the plot of \( \hat{W} \) rows projected onto the first two principal components) suggest that \( r = 5 \) and \( m = 2 \) lead to a most interpretable model fit. Then we apply K-means clustering on the rows of \( \hat{W} \) with \( m = 2 \) clusters and report the result in Table 1. It is interesting to observe that bio-related daily products including animal & animal products, vegetable products, over half of foodstuffs fall into cluster 1, most of which are all products of low durability. On the other hand, most industrial products including main parts of chemicals & allied industries, plastic/rubbers, stone/glass, machinery/electrical, aircraft, spacecraft, optical, photographic, etc., and clocks and watches constitute cluster 2.
Figure 1. Simulation 1–1 for general hLSM: the convergence of projection error $\|U(t)U^T - n^{-1}U^*U^T\|_F^2$ (also for $V, W$ resp.).

Figure 2. Simulation 1–2 for general hLSM: the convergence of the logarithm of sum of squares of projection error for $U(t), V(t), W(t)$ under different scales $\sigma \in \{1, 0.5, 0.1\}$.

Figure 3. Simulation 1–3 for general hLSM: the sum of squares of projection error for $U(t), V(t), W(t)$ and relative error defined as $||\hat{\Theta} - \Theta^*||_F / ||\Theta^*||_F$ with varying network sparsity.

Based on the layers clustering in Table 1, we further investigate the shared trade pattern among different countries/regions. Specifically, we construct a sub-tensor of size $48 \times 48 \times 20$ from cluster 1 for bio-related commodities, and a sub-tensor of size $48 \times 48 \times 16$ from cluster 2 for industrial commodities. A scientifically interesting question is to compare the latent position representations in these two groups of layers. Toward this end, we apply Algorithm 1 with $r = 3$ and $m = 1$ on these two sub-tensors. Since the trading flows are directed, the left singular vectors $\hat{U}$ and right singular vectors $\hat{V}$ are distinct.
Table 1. Network clustering results of 97 commodity layers, % denote the proportion of number of layers in the same category characterized by HS Code.

| Commodity cluster 1 | 01–05 Animal & Animal Products (100%) | 06–15 Vegetable Products (100%) | 16–18,23–24 Foodstuffs (56%) | 26 Mineral Products (33%) | 31,36–37 Chemicals & Allied Industries (27%) | 41,43 Raw Hides, Skins, Leather & Furs (66%) | 45–47 Wood & Wood Products (50%) | 50–55,57–58,60 Textiles (64%) | 66–67 Footwear / Headgear (50%) | 75,78–81 Metals (45%) | 86,89 Transportation (50%) | 92,93,97 Miscellaneous (37.5%) |
|---------------------|---------------------------------------|---------------------------------|-------------------------------|--------------------------|--------------------------------------------|-----------------------------------------|---------------------------------|---------------------------------|---------------------------------|-------------------------------|-------------------------------|-------------------------------|
| Commodity cluster 2 | 19–22 Foodstuffs (44%) | 25,27 Mineral Products (67%) | 28–30,32–35,38 Chemicals & Allied Industries (73%) | 39–40 Plastics / Rubbers (100%) | 42 Raw Hides, Skins, Leather, & Furs (33%) | 44,48–49 Wood & Wood Products (50%) | 56,59,61–63 Textiles (36%) | 64,65 Footwear / Headgear (50%) | 68–71 Stone / Glass (100%) | 72–74,76,82–83 Metals (55%) | 84–85 Machinery / Electrical (100%) | 87–88 Transportation (50%) | 90–91,94–96,99 Miscellaneous (62.5%) |

Figure 4. Latent positions of countries/regions for layers of (a) bio-related daily products (b) industrial products. Dataset: international trade flow.

![Figure 4](image)

Figure 5. ROC curve for link prediction on trade-flow dataset with 99.9% CI (red dashed line).

![Figure 5](image)

It turns out that the latent position of imports $\hat{V}$ provide clear and interpretable results. We further perform multidimensional scaling (MDS) on the rows of $\hat{V}_{\text{bio}}$ and $\hat{V}_{\text{ind}}$, projecting them into $\mathbb{R}^2$ for visualization. In Figure 4(a) and (b) plot the projected embedding of countries/regions according to their latent positions $\hat{V}_{\text{bio}}$ and $\hat{V}_{\text{ind}}$ after MDS, with nodes being colored by corresponding continents.

The latent positions of countries/regions for the two groups of layers exhibit different patterns. In Figure 4(a), latent positions for countries/regions in the same continent in general are close to each other, which to some extent reflects geographic proximity relations. We could observe several “clusters” such as European countries in the bottom right and the top middle; Hong Kong SAR, Singapore, South Korea (three out of Four Asian Tigers) and Japan in the bottom middle; Indonesia, Philippines, Thailand, and Malaysia (known as Tiger Cub Economies). This is reasonable since for commodities of low durability, regional trade partnerships usually dominate the inter-continental ones. However, it is interesting to notice those outliers. Three large economies China, United States, and Canada are relatively close in latent positions even though China is not geographically close to United States and Canada, since they export a large amount of bio-related/daily products to all other countries. Three South America countries (Argentina, Chile, and Brazil), two Africa countries (Nigeria and South Africa) and Mexico are embedded closer to the Middle East countries, as these nations import similar products mainly from several largest exporting economies. In Figure 4(b), the geographical impact, to some extent, is weakened. Germany, originally near United Kingdom, France, and Netherlands in Figure 4(a), is now clustered closer to China and United States, largely due to the fact that they are all big industrial nations with a huge demand of importing industrial raw materials. Denmark, Austria and Sweden (three out of Frugal Four) are mixed with countries/regions from Asia, South America and Africa, indicating that these developed industrial nations are heavily depending on imported industrial products from those developing countries. Australia is relocated nearer to European nations, which can be explained by their similarity of imported goods which outweighs the geographical closeness to Asian nations. Overall, high durability for industrial products...
means relatively low cost in freight and hence the trade partnerships are less regionally restricted and more related to their resemblance and connection in terms of industrial products.

To further assess the performance of the estimated latent positions, we apply our method to this dataset for the task of link prediction. We adopt the evaluation metric for link prediction in Zhao et al. (2017). Specifically, we set 20% of entries of $A$ (10% randomly selected out of nonzero entries and 10% out of zero entries) to be 0 and construct the test tensor data $A_{test}$. Then Algorithm 1 is applied to $A_{test}$ to get the estimated probability tensor $\hat{P} = g(\hat{\Theta})$. We evaluate the link prediction performance on those randomly deleted entries by AUC, which is defined to be the area under the ROC curve. By 30 simulations, we observe $AUC = 0.910(\pm 0.001)$. The ROC curve with 99.9% confidence interval is displayed in Figure 5.

6.2. Disease Hypergraph Network from MEDLINE

In the second data example, we analyze a hypergraph originated from the MEDLINE Database (www.nlm.nih.gov/medline). The database contains more than 27 million papers indexed by Medical Subject Headings (MeSH) concentrated on biomedicine. We focus on 12,637 papers published in 1960 annotated with 318 MeSH terms categorized into two types: Neoplasms (C04) and Nerve System Diseases (C10). In the constructed hypergraph network, the nodes are MeSH terms, and the hyperedges of sizes 1, 2, 3, 4, 5 are formed among the nodes annotated by the same paper. For simplicity, we only deal with triadic relations. We remove the hyperedges of size 1 and greater than 3, and add one additional dummy node for those hyperedges of size 2. We further abandon nodes of degrees less than 4 to eliminate those with insignificant information. Finally, we obtain an adjacency tensor $A$ sized $166 \times 166 \times 166$ (including one dummy node) of the hypergraph network with $n = 165$ MeSH terms, among which, 115 fall into class C04 and 50 are in class C10.

We run Algorithm 1 on $A$ and obtain the estimated $\hat{U}$ positions, in which, we set $r = 5$. Similarly, we perform MDS on $\hat{U}$ for visualization. The result of node embedding is plotted in Figure 6(a)–(b). Started with an initialization $U^{(0)}$ in Figure 6(a), where the two types of disease are mixed together, the eventual estimation $\hat{U}$ in Figure 6(b) shows a clear separation between the two clusters. Indeed, K-means clustering on the rows of $U^{(0)}$ and $\hat{U}$ with $K = 2$ clusters would produce 49.7% and 3.64% misclassification error rates, respectively. This clearly demonstrates the effectiveness and utility of our algorithm.

Finally, run a link prediction similar to that described in Section 6.1. Since the MeSH network is extremely sparse (99.9% entries of $A$ are 0’s), we construct the test tensor $A_{test}$ by randomly setting half of 1’s and the same number of 0’s to be 0, on which spots the accuracy of link prediction will be evaluated. This set up is constructed toward a balanced share between 0/1 values and a numerically stabler evaluation. Also in light of the observed sparsity, we choose a smaller scale parameter $\sigma$ in the link prediction here. We obtain $AUC = 0.944(\pm 0.005)$ over 30 simulations. The ROC curve with 99.9% confidence interval is presented in Figure 7.

7. Concluding Remarks

In this article, we propose a novel unified method for investigating the higher-order interactions in network data. Our framework is general in its abstraction of the concept “layer,” which could be either a third participant in a dyadic relationship/interaction, or it could index the multiple interactions between two nodes, or encode the time stamp in a dynamic network setting. Our model also allows the data generation scheme to connect to the interaction latent positions via a generalized linear link function. It covers several popular mainstream higher-order network models, including multilayer networks,
hypergraphs and dynamic networks, as special cases. Our proposed method is therefore versatile and widely applicable. Further, we developed original theory that rigorously guarantees the good performance of the algorithm and quantitatively understand the finite-sample error bounds over our method's iterations. In the following, we make several remarks on our method.

Network sparsity. Network sparsity $\mathbb{E}[\mathcal{E}|(n_1n_2n_3)^{-1}]$ plays a fundamental role in statistical theory (Ke, Shi, and Xia 2019; Jing et al. 2021). Unfortunately, LSM has its limitation in modeling network sparsity. The mainstream of latent factor models Hoff, Raftery, and Handcock (2002) / latent space models Zhang, Xue, and Zhu (2020) and MacDonald, Levina, and Zhu (2020) are mostly investigated without a network sparsity parameter except random dot product graph (with link function $g(x) = x$) (Athreya et al. 2017). Recently, Ma, Ma, and Yuan (2020) and Zhen and Wang (2021) propose an inner product sparse model ($\mathbf{C}^*$ is an identity tensor), but the statistical theory is only established for the globally optimal solution. It is unclear whether their estimators are solvable by any efficient algorithms. The main difficulty in addressing sparsity under our framework comes from the combination of (a) the nonlinear link function and (b) analyzing an iterative algorithm. Indeed, it is plausible to account for network sparsity in our model by, for instance, assuming $\max_{i,j,k,l} |\Theta|_{i,j,k,l} \leq -\theta$ for some large $\theta > 0$ (or considering $g(x) = \tau (1 + e^{-x})^{-1}$ for some $\tau \to 0$). However, such a treatment unavoidably complicates the constants $\gamma_\theta$ and $\beta_\tau$ (defined in Assumption 1) that further entangle with the sparsity parameter $\theta$ (or $\tau$), rendering it more difficult for isolating $\theta$ (or $\tau$) explicitly in the convergence analysis of our proposed algorithm. In contrast, we focus on the convergence and statistical performance of Algorithm 1, which becomes strikingly more challenging with an additional sparsity factor. Nevertheless, satisfactory simulation results can be delivered for sparse networks. See Section 5.1.

Choice of tuning parameters. The convergence of Algorithm 1 only requires $\delta_k$ to dominate $\mu_0(r_k/n_k)^{1/2}$. In practice, we can initially set $\delta_k = (\log(n_k)r_k/n_k)^{1/2}$ and gradually double it if Algorithm 1 fails to converge. Since $\Theta$ output by Algorithm 2 guarantees a good initialization, we set $\xi \sim \mathcal{N}(\hat{\Theta}|\|\hat{\Theta}\|_F)$ in practice. The learning rate should be chosen as $\eta = \eta_0 k_0^{-4} \Delta^{-2}(\mathbf{C}^*)$ according to Theorem 1. Similarly, we use the core tensor of $\hat{\Theta}$ to get an estimate of $\kappa_0$ and $\Delta^{-2}(\mathbf{C}^*)$. We find that it’s safe to set $\kappa_0 = 1$ in practice, as the dependency on condition number might be sub-optimal and we believe this is rather an artifact of our proof technique than a fundamental condition. It suffices to determine $\eta_0$, for which we can initially set it as, say, 10 and halve it if Algorithm 1 fails to converge.

Unknown rank. Algorithm 1 needs rank $\mathbf{r} = (r_1, r_2, r_3)$, which in practice is often unknown. With slight abuse of notation, denote $\ell_n(\mathbf{r})$ the likelihood value $\ell_n(\hat{\Theta})$, where $\hat{\Theta}$ is the output of Algorithm 1 with parameter $\mathbf{r}$. We adopt the Bayesian information criterion (BIC) to choose the rank. In particular, we estimate rank by $\hat{\mathbf{r}} = \arg \min_{\mathbf{r} \in \mathcal{R}} \left[ 2\ell_n(\mathbf{r}) + K_3 \log(n_1n_2n_3) \right]$, where $K_3 = r_1r_2r_3 + \sum_{k=1}^{3} [n_k r_k - 0.5 n_k (r_k + 1)]$ and $\mathcal{R}$ is the parameter space for $\mathbf{r}$. We are typically interested in low-rank setting and shall restrict $\mathcal{R}$ to a small set, say, $\mathcal{R} = \{ (r_1, r_2, r_3) : r_k \leq 10, r_k \in \mathbb{N}, k \in [3] \}$. For instance, this criterion selects $\hat{\mathbf{r}} = (5, 5, 5)$ in the MEDLINE dataset.

Computational complexity. In Algorithm 1, computing gradient takes $O(n_1n_2n_3)$, the SVD step takes $O \left( \sum_{k=1}^{3} n_k r_k^2 \right)$, and updating the core tensor $\mathbf{C}^{(i)}$ takes $O(n_1n_2n_3r_1r_2r_3 \cdot T^{(i)})$, where $T^{(i)}$ is the number of iterations needed by Newton’s algorithm. The latter step, as discussed in Section 5.1, is the main computation burden and we can use a random sampling procedure with sampling proportion $\rho_k$ to reduce its cost to $O(n_1n_2n_3r_1r_2r_3 \cdot \rho_k T^{(i)})$. Therefore, the per-iteration computational complexity of Algorithm 1 is $O(n_1n_2n_3r_1r_2r_3 \cdot T^{(i)})$ and that of accelerated version of Algorithm 1 is $O(n_1n_2n_3 \frac{1}{\rho})$ as long as $\rho_k T^{(i)} \cdot r_1r_2r_3 / \rho = O(1)$. In this case, the accelerated version has a comparable computational complexity with Zhang, Xue, and Zhu (2020), Ke, Shi, and Xia (2019), Cai, Li, and Xia (2021), and Han, Willett, and Zhang (2020).

Future work. There are a number of interesting directions of future work. In this work, we focused on binary network interactions. We expect our algorithm and analysis can be expanded to some weighted edge generation schemes, such as exponential distribution and some sub-Gaussian distributions. But given the volume of work even under the Bernoulli model, we stick to binary edges in this article and leave the direction for future investigation. Second, we constrain our data generation scheme to generalized linear link functions. While this formulation recently caters to the need of many real-life data analysis tasks, it is interesting to expand the methodology to more general link functions. A third interesting but much more challenging future exploration is to account for dependency between the higher-order interactions.

Supplementary Materials

Supplement to “Latent Space Model for Higher-order Networks and Generalized Tensor Decomposition”: The material includes more simulation results, technical lemmas and detailed proofs of the theoretical results in the main article. (supp.pdf, pdf file)

Codes: The material includes the R codes and data needed to reproduce empirical results in this article. Please see README in the zip file for more information. (codes.zip, zip archive)

Acknowledgments

The authors would like to thank the Associate Editor and reviewers for careful reading, insightful comments and constructive suggestions to help improve the quality of the article.

Disclosure Statement

The authors report there are no competing interests to declare.

Funding

Lyu and Xia’s research is supported in part by Hong Kong RGC Grant GRF 16303320 and GRF 16300121.
Z. LYU, D. XIA, AND Y. ZHANG

(2020b), “Spectral and Matrix Factorization Methods for Consistent Community Detection in Multi-Layer Networks,” *The Annals of Statistics*, 48, 230–250. [1321, 1328]

Pensky, M. (2019), “Dynamic Network Models and Graphon Estimation,” *Annals of Statistics*, 47, 2378–2403. [1321]

Pensky, M., and Zhang, T. (2019), “Spectral Clustering in the Dynamic Stochastic Block Model,” *Electronic Journal of Statistics*, 13, 678–709. [1321]

Rohe, K., Chatterjee, S., and Yu, B. (2011). Spectral clustering and the high-dimensional stochastic blockmodel. *Annals of Statistics*, 39(4):1878–1915. [1321]

Sarkar, P., and Moore, A. W. (2005), “Dynamic Social Network Analysis using Latent Space Models,” *Acm Sigkdd Explorations Newsletter*, 7, 31–40. [1325]

Sewell, D. K., and Chen, Y. (2015), “Latent Space Models for Dynamic Networks,” *Journal of the American Statistical Association*, 110, 1646–1657. [1325]

Tang, R., Tang, M., Vogelstein, J. T., and Priebe, C. E. (2017), “Robust Estimation from Multiple Graphs Under Gross Error Contamination,” arXiv preprint arXiv:1707.03487. [1320]

Wang, D., Yu, Y., and Rinaldo, A. (2018), “Optimal Change Point Detection and Localization in Sparse Dynamic Networks,” arXiv preprint arXiv:1809.09602. [1321, 1325, 1329]

Wang, H., Tang, M., Park, Y., and Priebe, C. E. (2013), “Locality Statistics for Anomaly Detection in Time Series of Graphs,” *IEEE Transactions on Signal Processing*, 62, 703–717. [1321]

Wang, L., Zhang, Z., and Dunson, D. (2019), “Common and Individual Structure of Brain Networks,” *The Annals of Applied Statistics*, 13, 85–112. [1320]

Wang, M., and Li, L. (2020), “Learning from Binary Multiway Data: Probabilistic Tensor Decomposition and its Statistical Optimality,” *Journal of Machine Learning Research*, 21, 1–38. [1320, 1325]

Wang, Y., Chakrabarti, A., Sikavoff, D., and Parthasarathy, S. (2017), “Fast Change Point Detection on Dynamic Social Networks,” arXiv preprint arXiv:1705.07325. [1321]

Wilson, J. D., Stevens, N. T., and Woodall, W. H. (2019), “Modeling and Detecting Change in Temporal Networks via the Degree Corrected Stochastic Block Model,” *Quality and Reliability Engineering International*, 35, 1363–1378. [1321]

Xia, D. (2019), “Normal Approximation and Confidence Region of Singular Subspaces,” arXiv preprint arXiv:1901.00304.

Xia, D., and Yuan, M. (2019), “On Polynomial Time Methods for Exact Low-Rank Tensor Completion,” *Foundations of Computational Mathematics*, 19, 1265–1313. [1322, 1325, 1330]

Xu, K. (2015), “Stochastic Block Transition Models for Dynamic Networks,” in *Artificial Intelligence and Statistics*, pp. 1079–1087, PMLR. [1325]

Yuan, M., Liu, R., Feng, Y., and Shang, Z. (2018), “Testing Community Structures for Hypergraphs,” arXiv preprint arXiv:1810.04617. [1321, 1324, 1329]

Zhang, A., and Xia, D. (2018), “Tensor SVD: Statistical and Computational Limits,” *IEEE Transactions on Information Theory*, 64, 7311–7338. [1330]

Zhang, J., and Cao, J. (2017), “Finding Common Modules in a Time-Varying Network with Application to the Drosophila Melanogaster Gene Regulation Network,” *Journal of the American Statistical Association*, 112, 994–1008. [1320, 1328]

Zhang, X., Xu, G., and Zhu, J. (2021), “Joint Latent Space Models for Network Data with High-Dimensional Node Variables,” *Biometrika*, 109, 707–720. [1323]

Zhang, X., Xue, S., and Zhu, J. (2020), “A Flexible Latent Space Model for Multilayer Networks,” in *International Conference on Machine Learning*, pp. 11288–11297, PMLR. [1321, 1322, 1323, 1324, 1328, 1334]

Zhang, Y., Levin, E., and Zhu, J. (2016), “Community Detection in Networks with Node Features,” *Electronic Journal of Statistics*, 10, 3153–3178. [1321]

(2020), “Detecting Overlapping Communities in Networks using Spectral Methods,” *SIAM Journal on Mathematics of Data Science*, 2, 265–283. [1321]

Zhao, Y., Wu, Y.-J., Levin, E., and Zhu, J. (2017), “Link Prediction for Partially Observed Networks,” *Journal of Computational and Graphical Statistics*, 26, 725–733. [1333]

Zhao, Z., Chen, L., and Lin, L. (2019), “Change-Point Detection in Dynamic Networks via Graphon Estimation,” arXiv preprint arXiv:1908.01823. [1329]

Zhen, Y., and Wang, J. (2021), “Community Detection in General Hypergraph via Graph Embedding,” arXiv preprint arXiv:2103.15035. [1321, 1322, 1334]