Functional edged network model based on functional tensor decomposition

Anonymous Author
Anonymous Institution

Abstract

This paper aims at modeling functional network with irregular observation. Compared with existing functional network model, we focus on the functional edges and use tensor to express the data. To deal with the high-dimensional functional edge and the community structure problem, we proposed the functional edges network(FEN) model. FEN model can express the spatial information with symmetrical matrix bases, community structure, and increase the the interpretability of the model. We also use Riemann conjugate gradient descent optimization to estimate the FEN model with irregular observation to directly get functional edges with smoothness constraints. We use simulation data and the real subway data of Hong Kong to verify the advantages of FEN model. Then we proposed some theorems to guarantee the properties of FEN model.

1 INTRODUCTION

Graphical Model are widely used in modeling the relationships between different variables. Usually, the weight of each edge which represents their connection strength is not a constant value, but a value changing over time. Most of current methods assume the weights of all the edges are observed at regularly spaced discrete time points, and use dynamic graphical models(Durante et al., 2017; Loyal and Chen, 2021). This assumption, however, is often unsatisfied when the observation points are irregular. In these cases, it is more suitable to regard the dynamic weight over time as a continuous function and the observations are sampled from the function at certain discrete time points. In this way, we can observe or infer the edge value at any time with flexibility.

Preliminary work. Under review by AISTATS 2023. Do not distribute.

Similarly to the adjacency matrix of the two-dimensional network, we use a three-dimensional functional tensor where the third dimension is continuous functions to express the functional-edged network.

Furthermore, the network should have the community structure. The community structure is the topological structure which clusters nodes into different groups. If we can find the structure, we can use it to decrease the dimension of our model, increase the spatial interpretability and do some data analysis better, like clustering. The community structure has been used in many researches of transport system(Zhong et al., 2015; Xu et al., 2016). So we need to find the most reasonable structure to do the modeling.

Though there are emerging works considering functional network modeling, all of them consider the node data are functions, and the probabilistic edges describe the dependence structure of nodes. So far to our best knowledge, there is no work dealing with functional-edged network. To fill this research gap, we proposed a Functional Edged Network(FEN) model. In particular, we propose a novel functional tucker decomposition, which can on the one hand extract features for high-dimensional functional data with irregular sampling points, and on the other can describe community structure of the network. In particular, our FEN model has the following contributions: 1) we extract the community information in the adjacency functional tensor with symmetrical matrix factorization; 2) we conduct functional decomposition to extract smooth dynamic patterns of the community structure; 3) We can deal with irregularly sampled data with kernel smoothing as preprocessing; 4) We proposes an efficient model estimation algorithm with Riemann conjugate gradient descent optimization approach.

2 LITERATURE REVIEW

If we neglect the spatial structure of the network, we can regard the $m(m-1)$ functional edges as multivariate functions, for which methods based on functional principal component analysis (FPCA) are most commonly used for modeling.

For functional data with regular sampling points, traditional PCA can be revised and applied for modeling, like

vectorized PCA (VPCA)(Nomikos and MacGregor, 1995) and multivariate FPCA (MFPCA)(Paynabar et al., 2016). There are also some other settings of FPCA like smoothed FPCA(Foutz and Jank, 2010), robust FPCA(Bali et al., 2011). If we want to deal with irregular sampling data, Yao et al. (2005) is a suitable method which use kernel smoothing algorithm to estimate the covariance function for PCA. Later, Greven et al. (2011) improved it to more strict settings, they regarded the time of observe points as the random variables with a given distribution.

All these methods cannot address network structure, to solve this problem, we need to consider the functional network analysis(Qiao et al., 2019, 2020; Zhu et al., 2016). There are two main model of this methods, precision matrix(the inverse of the covariance matrix) estimation(Qiao et al., 2019; Chun et al., 2013; Danaher et al., 2014) and the nerghborhood-based estimation(Kolar and Xing, 2011). The latter can be further divided into three different classes: 1) graphical structure changes with functional-type (Zhou et al., 2010); 2) the data changes with functional (Qiao et al., 2019; Zhu et al., 2016); 3) both of the graphical structure and data have functional changes(Qiao et al., 2020). For the first class considering functional changed graphical structures, if the data are regularly sampled, it can be also regarded as dynamic networks for modeling. The latent space model(LSM) is perfect to modeling the dynamic network with community structure, since we can regard each latent state as a community and find the possibility that each node belongs to each community(Sewell and Chen, 2017; Robinson and Priebe, 2012). Besides all of this, Rogers et al. (2013) also improved the traditional linear dynamic system model (LDS), propose multi-linear dynamic system model (MLDS) ro modeling the time series tensor which is similar to our model setting. Most of the research methods are for the "probability edge", which means they only think node set is real, with the probability of the existence of the matrix to represent the edge precision, so for edge set there is not a direct estimation. In our method, we prefer to directly estimate the edge set.

If we neglect the smoothness constraints, we can use some tensor decomposition algorithms to complete the adjacency tensor, like Tucker decomposition (Kolda and Bader, 2009), CANDECOMP/PARAFAC(CP) decomposition(Goulart et al., 2015), Tensor Train(Oseledets, 2011), Tensor Ring(Liu et al., 2021) and Tensor Network(Batselier et al., 2017). But in this way, we may lose a lot of estimation accuracy because the third dimension of our adjacency matrix includes some continuous smooth functions.

In the case of two dimensional tensors, i.e., matrix, there are some works considering smoothness of decomposition. McNeil et al. (2021) proposed a decomposition algorithm for matrix decompsotion where one dimension is assumed to represent continuous time. For three dimensional tensors, Yokota et al. (2016) use CP decomposition to model the tensor network and complete missing data. They use the differential matrices and smoothing coefficients to constrain the smoothness of edges in the network. In the field of video recovery Hu et al. (2015) improved the t-SVD decomposition and proposed the twist tensor nuclear norm(t-TNN) algorithm which decomposed and completed the tensor data by twisting to achieve the effect of smooth constraints and effectively exploit the temporal redundancy between frames.

In summary, so far to our best knowledge, there is no work that considers functional data as edges in network analysis. Yet the above methods motivate us to use tensor analysis to describe the functional network, which is also the innovation of our research.

3 FEN MODEL

3.1 Model formulation

We assume the functional network has m nodes and at most $m(m-1)$ edges. The weight value of each edge can variate with time. More specifically, we assume that edge between node i and j has a weight function with respect to time as $X_{ij}(t)$.

Then we can denote the adjacency tensor $\mathcal{X} \in \mathbb{R}^{m\times m\times \mathcal{T}}$ as a three dimensional functional tensor which consists of $m(m-1)$ weight functions as described above. The third dimension of $\mathcal{X}$ is continuous and represents weight values over time $\mathcal{T} = [T_s, T_e]$ which is a closed interval on $\mathbb{R}$. In this way, we can use $\mathcal{X}(t)$ to record the adjacency matrix at any time t.

To describe the community structure in the network, for the third dimension, time dimension, we can do the decomposition as following,

$$\begin{aligned} X_{ij}(t) &= \sum_{k=1}^{s}\sum_{l=1}^{s} \phi_{ik} c_{kl}(t) \phi_{jl} \\ s.t. \quad & \Phi^T\Phi = \mathbf{I}_s \end{aligned} \tag{1}$$

where $\mathbf{I}_s$ denotes the Identity matrix of order s. Assume the number of community, s, is pre-specified. For each element $\phi_{ik} \in \Phi$, it can be interpreted as the possibility that node i belongs to community k and $0 \le \phi_{ik}^2 \le 1$. The higher value of $|\phi_{ik}|$, the more likely node i is in community k, and the sign of ϕ_{ik} denotes the attitude of node i to community k. We use $c_{kl}(t)$ denotes the weight function between community k and community l.

To further describe the temporal features of $c_{kl}(t)$, we assume $c_{kl}(t)$ as a linear combination of K functional bases $\{\vartheta_k(t), k = 1, \ldots, K\}$ as:

$$X_{ij}(t) = \sum_{a=1}^{s}\sum_{b=1}^{s}\sum_{k=1}^{K} b_{abk}\phi_{ia}\phi_{jb}\vartheta_k(t) \tag{2}$$

Equation 1 can be rewritten in the tensor form as:

$$\mathcal{X} = \mathbf{B} \times_1 \Phi \times_2 \Phi \times_3 \Theta \tag{3}$$

where $\mathbf{B} \in \mathbb{R}^{s\times s\times K}$, $\Phi \in \mathbb{R}^{m\times s}$ and $\Theta \in \mathbb{R}^{\mathcal{T}\times K}$. s and K are the hyperparameters that need to be determined in advance. For convenience, we think the $\mathcal{X}$ has been centralized and do not consider the decomposition with mean function.

Consider the observation noise, we can define the observe functions as following

$$\mathcal{Y} = \mathcal{X} + \mathcal{E} = \mathbf{B} \times_1 \Phi \times_2 \Phi \times_3 \Theta + \mathcal{E} \tag{4}$$

where $\mathcal{Y} \in \mathbb{R}^{m\times m\times \mathcal{T}}$ denotes the observe function with noise function $\mathcal{E} \in \mathbb{R}^{m\times m\times \mathcal{T}}$. Each point in $\mathcal{E}$, $\epsilon_{ij}(t), i = 1, \cdots, m \quad j = 1, \cdots, m \quad t \in \mathcal{T}$ obeys normal distribution with mean of 0 and variance of σ^2 and is independent with each other point.

Equation 4 expresses the observation function when we only have one sample, if we have multiple samples, we need to introduce a fourth dimension and the data become $\mathcal{X}, \mathcal{Y}, \mathcal{E} \in \mathbb{R}^{m\times m\times \mathcal{T}\times N}$ where N represents the number of samples. However, the fourth dimension do not have decomposition, so we have

$$\mathcal{Y} = \mathbf{B} \times_1 \Phi \times_2 \Phi \times_3 \Theta \times_4 I + \mathcal{E} \tag{5}$$

where $I \in \mathbb{R}^{N\times N}$ is an identity matrix.

3.2 Model inference

Now we introduce the model estimation framework. According to our problem setting, we need the weight function on each edge is a smooth function, then we have to add the smooth constraints when we do the estimation. We want to use l_2 loss as objective function. Combine with the tucker decomposition, we can add the smoothing constraints to the third continues basis matrix to guarantee the smoothness of the weight function. Then we can rewrite our FEN model as solving the following optimization problem

$$\begin{aligned}[\hat{\mathcal{X}}, \hat{\mathbf{B}}, \hat{\Phi}, \hat{\Theta}] = &\underset{\mathcal{X},\mathbf{B},\Phi,\Theta}{\arg\min} \frac{1}{2}||\mathcal{Y} - \mathcal{X}||_F^2 + \\ &\sum_{k=1}^{K} \alpha_k \int_{\mathcal{T}} (\vartheta_k{}'(t))^2\, dt \\ s.t. &\mathcal{X} = \mathbf{B} \times_1 \Phi \times_2 \Phi \times_3 \Theta \\ &\Phi^T\Phi = \mathbf{I}_s \quad \Theta^T\Theta = \mathbf{I}_K \quad \vartheta_k \in \mathbf{C}^1\mathcal{T}\end{aligned} \tag{6}$$

where $||a||_F^2$ = is the F norm for continues dimension defined as $\mathcal{Y} \in \mathbb{R}^{m\times m\times \mathcal{T}\times N}$ and $||\mathcal{Y}||_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{n=1}^{N}\int_{\mathcal{T}} Y_{ijn}(t)^2\, dt}$. In this way, we can calculate the F norm in Equation 6. ϑ_k denotes the kth continues basis function of Θ and $\mathbf{C}^1\mathcal{T}$ denotes the set which consists of all first order continuous differentiable function on $\mathcal{T}$. In this way, we guarantee functions in the tensor $\mathcal{X}$ is smooth differentiable by limiting smooth differentiable functions in the continues basis matrix Θ.

We need to note that, the above optimization problem contain the continues function. If we can actually observe the hole function, we can solve the FEN model. But in reality, each edge $Y_{ijn}(t)$ are actually observable at a series of discrete time points, which may be irregular. To make uniform notation, we choose the smallest sampling time resolution of all the edges as the global sampling time resolution, and define a set of observation points $\tilde{\mathcal{T}} = \{T_s + \frac{l}{L}(T_e - T_s) | l = 1, 2, \ldots, L\}$. Since L is the upper bound of sampling frequency of all the edges, any observation points can be the subset of $\tilde{\mathcal{T}}$. On the one hand, L can go infinity so any observation points can be the subset of $\tilde{\mathcal{T}}$. On the other hand, the sensors observed in real life have a certain sampling frequency. In this way, the isometric segmentation is more reasonable.

Then the lth observation of edge between node i and node j can be written as follow

$$\begin{aligned}Y_{ijn}(T_l) &= X_{ijn}(T_l) + \epsilon_{ijn}(T_l) \\ &= \sum_{a=1}^{s}\sum_{b=1}^{s}\sum_{k=1}^{K} b_{abkn}\phi_{ia}\phi_{jb}\vartheta_k(T_l) + \epsilon_{ijn}(T_l)\end{aligned} \tag{7}$$

where the $\epsilon_{ijn}(T_l)$ is independent random variables which obay the normal distribution with mean of 0 and variance of σ^2. $T_l = T_s + \frac{l}{L}(T_e - T_s) \in \tilde{\mathcal{T}}$.

Suppose we can observe edges at all the time points of $\tilde{\mathcal{T}}$, we can get the full observe discrete tensor as following:

$$\mathbf{Y}^F = \mathbf{X} + \mathbf{E} = \mathbf{B} \times_1 \Phi \times_2 \Phi \times_3 \Theta + \mathbf{E} \tag{8}$$

where $\mathbf{X} \in \mathbb{R}^{m\times m\times L\times N}, \Theta \in \mathbb{R}^{L\times K}$ denote dicretilized data at the corresponding values of $\tilde{\mathcal{T}}$. $\mathbf{Y}^F \in \mathbb{R}^{m\times m\times L\times N}$ denotes the full observation in the observation points set, and $\mathbf{E} \in \mathbb{R}^{m\times m\times L\times N}$ consists of independent and identically distributed normal random variable noise with mean of 0 and variance of σ^2.

In reality, because the data are actually sampled at irregular places, we cannot observe the full observation tensor $\mathbf{Y}^F$, and the observable points of different functions are acturally not the same. As such we use mask tensor to express the irregular observation. We define the mask tensor $\Omega \in \mathbb{R}^{m\times m\times L\times N}$ whose value is 1 at the points that $\mathbf{Y}^F$ can be observed and 0 at the points that $\mathbf{Y}^F$ can not be observed. Then we have

$$\mathbf{Y} = \Omega * \mathbf{Y}^F \tag{9}$$

where the $\mathbf{Y}$ is the observation tensor we can get. We will use it to estimation $\mathcal{X}, \mathbf{B}, \Phi, \Theta$. The $*$ denotes the Hadamard product.

Then we can combine the discrete observation $\mathbf{Y}$ and its decomposition to rewrite the Equation 6 as following:

$$\begin{aligned}[\hat{\mathbf{X}}, \hat{\mathbf{B}}, \hat{\Phi}, \hat{\Theta}] = &\arg\min_{\mathbf{X},\mathbf{B},\Phi,\Theta} \frac{1}{2}||\mathcal{P}_\Omega(\mathbf{Y} - \mathbf{X})||_F^2 + \\ &\frac{1}{2}\sum_{k=1}^{K} \alpha_k \theta_k^T \mathbf{H} \theta_k \\ s.t. &\mathbf{X} = \mathbf{B} \times_1 \Phi \times_2 \Phi \times_3 \Theta \\ &\Phi^T \Phi = \mathbf{I}_s \quad \Theta^T \Theta = \mathbf{I}_K \end{aligned} \tag{10}$$

similarly to the continue case, we also add the smoothness constraints. $\mathbf{H}$ denotes differential matrix, θ_k denotes the kth column of Θ and α_k is the smoothing constraint coefficient. The $\mathcal{P}_\Omega(\cdot)$ denotes the observable points and set the unobservable points as 0. For example, $\mathcal{P}_\Omega(\mathbf{A}) = \Omega * \mathbf{A}$. Then we can get the $\hat{\mathbf{X}}$ and its decomposition which are the estimation of $\mathbf{X}$ and its decomposition. Then we can interpolate and complete the discrete $\hat{\mathbf{X}}$ in the third dimension as an estimation of $\mathcal{X}$. When the size of observation points set L is large, the interpolation and completion is not difficult, then we only consider the estimation of $\mathbf{X}$ and its decomposition in subsequent sections.

So far, we have completed the establishment of FEN model. By solving Equation 10, we can obtain a functional network model that satisfies the smoothness constraint, has the dimension reduction of community structure, and can be estimated by irregular observation data. In the next section, we will introduce how to solve the optimization problem.

4 OPTIMIZATION ALGORITHM

4.1 Low Rank Space and Riemann Manifold

In this section, we use conjugate gradient method under riemann optimization to solve the Equation 10, but we do not require the first two bases to be the same. In other words, we will first estimate the decomposition of $\mathbf{X}$ as $\mathbf{X} = \mathbf{B} \times_1 \Phi \times_2 \Psi \times_3 \Theta$. Accordingly, we need to add the the symmetry correction at the end of algorithm to make sure the first two bases are the same. For convenience, we use $\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3$ to denote Φ, Ψ, Θ in this section.

We define the low rank space $\mathcal{M}_\mathbf{r}$

$$\begin{aligned}\mathcal{M}_\mathbf{r} = \{&\mathbf{X} = \mathbf{B} \times_{i=1}^{4} \mathbf{U}_i | \\ &\mathbf{B} \in \mathbb{R}^{r_1 \times \cdots \times r_4}, \mathbf{U}_i \in \mathbb{R}^{n_i \times r_i}, \mathbf{U}_i^T \mathbf{U} = \mathbf{I}_{r_i}\}\end{aligned} \tag{11}$$

where the $\mathbf{I}_{r_i}$ denotes the r_i order identity matrix, $r_1, r_2, r_3, r_4 = s, s, K, N$ and $n_1, n_2, n_3, n_4 = m, m, L, N$.

It is not difficult to find that the target decomposition $\hat{\mathbf{X}} \in \mathcal{M}_\mathbf{r}, \mathbf{r} = [s, s, K, N]$, so we need to search the optimal solution in the low rank space $\mathcal{M}_\mathbf{r}$. The low rank space $\mathcal{M}_\mathbf{r}$ which defined by the tucker decomposition is an classical Riemann manifold. In this way, to solve the optimization problem, we need to use an optimization method under Riemann optimization. Among the traditional gradient-based optimization methods, conjugate gradient method considers both the computational complexity and convergence speed. So we use the conjugate gradient method under Riemann optimization to solve the objective problem. Then we will introduce this optimization method.

4.2 Conjugate Gradient Method under Riemann Optimization

We first give the algorithm framework of conjugate gradient method under Riemann optimization in Algorithm 1

Algorithm 1 Conjugate Gradient method under Riemann Optimization

Require: observe tensor $\mathbf{Y}$
mask tensor Ω
rank of tucker decomposition $\mathbf{r} = [s, s, K, N]$
smoothing constraint coefficient $\alpha_k, k = 1, \cdots, K$
differential matrix $\mathbf{H}$
tolerate error δ
initial value $\mathbf{X}_0 \in \mathcal{M}_\mathbf{r}$
Ensure: solution of Equation 10 $\hat{\mathbf{X}}, \hat{\mathbf{B}}, \hat{\Phi}, \hat{\Theta}$
$\eta_0 = -\mathrm{grad} f(\mathbf{X}_0)$
$\gamma_0 = \arg\min_\gamma f(\mathbf{X}_0 + \gamma\eta_0)$
$\mathbf{X}_1 = R(\mathbf{X}_0, \gamma_0\eta_0)$
$k = 1$
while not converge **do**
$\xi_k = \mathrm{grad} f(\mathbf{X}_k)$
$\eta_k = -\xi_k + \beta_k \mathcal{F}_{\mathbf{X}_{k-1} \to \mathbf{X}_k} \eta_{k-1}$
$\gamma_k = \arg\min_\gamma f(\mathbf{X}_k + \gamma\eta_k)$
$\mathbf{X}_k + 1 = R(\mathbf{X}_k, \gamma_k\eta_k)$
$k = k + 1$
end while
$\hat{\mathbf{X}} = \mathcal{D}(\mathbf{X}_{out})$

where $f(\mathbf{X}_k)$ is the objective function of Equation 10 where we replace $\mathbf{X}$ by $\mathbf{X}_k$. $\mathbf{X}_{out}$ is the last $\mathbf{X}_k$ which is the output of the while loop.

Under the framework of Riemann optimization, the optimization function is restricted to the Riemann surface, and the optimization method in the original Euclidian space needs to be carried out in the tangent plane of every point on the Riemann manifold. Then we will introduct the details of Algorithm 1.

For each point $\mathbf{X}$ on the Riemann manifold with parameter $\mathbf{r}$, we first define a concrete representation of its tangent plane $T_\mathbf{X}\mathcal{M}_\mathbf{r}$. Then we can map the gradient of objective

function in Euclidean space to the tangent plane. We call the gradient tensor in the tangent plane as the Riemann gradient, $\mathrm{grad} f(\mathbf{X}_k)$. Then we need to define a gradient transfer projection,$\mathcal{F}_{\mathbf{X}_{k-1}\to\mathbf{X}_k}$, to update the Riemann gradient in the tangent plane of the last point $\mathbf{X}_{k-1}$ to the tangent plane of the current point $\mathbf{X}_k$. Similarly with the traditional conjugate gradient method, we need to calculate the conjugate direction coefficient β_k and step size γ_k. Then we use HOSVD algorithm as the retraction, $R(\cdot)$, to map points on the tangent plane into manifold form,e.g. $R(\mathbf{X}_k, \gamma_k\eta_k) = HOSVD(\mathbf{X}_k + \gamma_k\eta_k)$. All the details of formulation of above are shown in supplement.

Through the above description, we can get a set of solution results of Riemann optimization, $\mathbf{X}_{out} = \tilde{\mathbf{B}} \times_1 \hat{\mathbf{U}}_1 \times_2 \hat{\mathbf{U}}_2 \times_3 \hat{\mathbf{U}}_3$, but it does not satisfy the symmetry constraint of the bases. Then we can use Algorithm 2 to obtain the object decomposition of symmetric bases in accordance with the constraints of Equation 10, $\hat{\mathbf{X}} = \mathcal{D}(\mathbf{X}_{out})$. It is to be noted that we need to keep $\mathbf{U}_4 = \mathbf{I}_N$ during the algorithm to guarantee there is no dimensional reduction in the sample dimension. So far, we have solved the Equation 10 and got the estimation of parameters in the FEN model $\hat{\mathbf{X}} = \hat{\mathbf{B}} \times_1 \hat{\Phi} \times_2 \hat{\Phi} \times_3 \hat{\Theta}$.

Algorithm 2 Base Symmetrization

Require: decomposition of asymmetric basis$\mathbf{X}_{out} = \tilde{\mathbf{B}} \times_1 \hat{\mathbf{U}}_1 \times_2 \hat{\mathbf{U}}_2 \times_3 \hat{\mathbf{U}}_3$

Ensure: decomposition of symmetric basis $\mathcal{D}(\mathbf{X}_{out}) = \hat{\mathbf{B}} \times_1 \hat{\mathbf{U}}_1 \times_2 \hat{\mathbf{U}}_1 \times_3 \hat{\mathbf{U}}_3$

$\mathbf{O} = \hat{\mathbf{U}}_2{}^T \hat{\mathbf{U}}_1$

$\hat{\mathbf{B}} = \tilde{\mathbf{B}} \times_2 \mathbf{O}^T$

$\mathcal{D}(\mathbf{X}_{out}) = \hat{\mathbf{B}} \times_1 \hat{\mathbf{U}}_1 \times_2 \hat{\mathbf{U}}_1 \times_3 \hat{\mathbf{U}}_3$

5 THEOREM

In this section, we derive the theoretical properties of FEN model and propose three main theorems and one important corollary. As Equation 10 with smoothing constraint is too complex, the discussion in this section is based on the premise that the smoothing constraint coefficient $\alpha_k = 0$.

5.1 Assumptions

After ignoring the smoothing constraint coefficient, we can rewrite Equation 10 in a more general formulation,

$$\hat{\mathbf{X}} = \underset{\mathbf{X}\in\mathcal{M}_{\mathbf{r}}}{\arg\min} ||\mathcal{P}_\Omega(\mathbf{Y} - \mathbf{X})||_F \tag{12}$$

where $\mathbf{Y} = \mathbf{X} + \mathbf{E}$ and $\mathbf{Y}, \mathbf{X}, \mathbf{E}, \Omega \in \mathbb{R}^{n_1\times\cdots\times n_4}$. In our case, $n_1 = n_2 = m, n_3 = L, n_4 = N$. Each element ϵ in tensor $\mathbf{E} \sim N(0, \sigma^2)$ and independent. We can find that Equation 12 does not have symmetry constraints, but we can always use Algorithm 2 to correct the basis without changing $\hat{\mathbf{X}}$. So we do not consider the symmetry constraints in the rest of this section. We treat $\hat{\mathbf{X}}$ as the estimation of $\mathbf{X}$ and we will talk about its properties.

First, we give the definition of rank of tensor

$$\mathbf{rank}_i(\mathbf{X}) = \mathbf{rank}(\mathbf{X}_{(i)}) \quad i = 1, \cdots, 4 \tag{13}$$

Assumption 1. *Define $R_i = \boldsymbol{rank}_i(\mathbf{Y})$, and*

$$r_i \le R_i \quad i = 1, \cdots, 4 \tag{14}$$

Since the hyperparameter $\mathbf{r}$ is a vector that we choose, we can always select the $\mathbf{r}$ that matches the Assumption 1 by slowly increasing $\mathbf{r}$ during the actual running of the algorithm . In this way, Assumption 1 can be achieved.

Assumption 2. *if $\mathbf{X}_1 \in \mathcal{M}_{\mathbf{r}_1}, \mathbf{X}_2 \in \mathcal{M}_{\mathbf{r}_2}$, then we have*

$$||\mathcal{P}_\Omega(\mathbf{X}_1 - \mathbf{X}_2)||_F \in [c||\mathbf{X}_1 - \mathbf{X}_2||_F, C||\mathbf{X}_1 - \mathbf{X}_2||_F] \tag{15}$$

where c, C are constants which are only related to mask tensor Ω.

Since $\mathbf{X}_1$ and $\mathbf{X}_2$ are in different low rank space $\mathcal{M}_{\mathbf{r}_1}$ and $\mathcal{M}_{\mathbf{r}_2}$ and they are obtained by the tucker product of different rank. In this way, $\mathbf{X}_1$ and $\mathbf{X}_2$ have uniform continuity property, that is, the norm at the unobserved point can be linearly constrained by its neighboring points.

5.2 Theorems

Theorem 1. *If there are two solution of Equation 12 satisfy $||\mathcal{P}_\Omega(\hat{\mathbf{X}}_1 - \mathbf{Y})||_F = ||\mathcal{P}_\Omega(\hat{\mathbf{X}}_2 - \mathbf{Y})||_F = \epsilon$, and under Assumption 2, there exists the same set of orthogonal basis $\mathbf{U}_1, \cdots, \mathbf{U}_d$, which makes the following statement true*

$$\begin{aligned} \hat{\mathbf{X}}_1 &= \mathbf{G}_1 \times_{i=1}^4 \mathbf{U}_i \\ \hat{\mathbf{X}}_2 &= \mathbf{G}_2 \times_{i=1}^4 \mathbf{U}_i \\ ||\mathbf{G}_1 - \mathbf{G}_2||_F &\le 2C\epsilon \end{aligned} \tag{16}$$

where C is the constant in Assumption 2, ϵ is very small which we'll prove in the following theorem.

Theorem 1 shows that even if there are two optimal solutions to Equation 12, the two solutions are same in F-norm without considering the orthogonality of the bases or its different order. This theorem ensures that our objective function is always identifiable.

Theorem 2. *Under Assumption 1 and Assumption 2, we have*

$$||\hat{\mathbf{X}} - \mathbf{X}||_F \le \frac{C}{c}(||\mathbf{E}||_F + \sqrt{\sum_{i=1}^4 \Lambda_i}) \tag{17}$$

where $\frac{||\mathbf{E}||_F^2}{\sigma^2} \sim \chi_{N_{tol}}, \frac{\Lambda_i}{\sigma^2} \sim noncentral - \chi(\lambda_i, (1 - \frac{r_i}{R_i})N_{tol}), \lambda_i \le (1 - \frac{r_i}{R_i})\frac{||\mathbf{X}||_F^2}{\sigma^2}$, $N_{tol} = \prod_{i=1}^4 n_i$ denotes

the number of elements in $\mathbf{Y}$*. Futher more*

$$\begin{aligned}\mathbb{E}(\Lambda_i) &= \sigma^2(\lambda_i + (1 - \frac{r_i}{R_i})N_{tol}) \\ &\leq (1 - \frac{r_i}{R_i})(||\mathbf{X}||_F^2 + \sigma^2 N_{tol})\end{aligned} \tag{18}$$

but $||\mathbf{E}||_F^2, \Lambda_1, \cdots, \Lambda_d$ *are not independent. In particular, when* $r_i = R_i \quad i = 1, \cdots, 4$*,* Λ_i *degenerates into a constant random variable with a value of 0.*

Theorem 2 characterizes an upper bound on the F-norm distance between our estimate $\hat{\mathbf{X}}$ and the true value $\mathbf{X}$, and expresses this upper bound by a combination of multiple chi-square distributions. As the hyperparameter $\mathbf{r}$ gets closer and closer to the rank $\mathbf{R}$, the expectation of this estimated upper bound distribution is getting smaller and smaller. In order to better characterize this upper bound, we give Corollary 1 as follows.

Corollary 1. *Under Assumption 1 and Assumption 2, according to Theorem 2, we can get*

$$P(||\hat{\mathbf{X}} - \mathbf{X}||_F \leq \epsilon) \geq$$
$$1 - \frac{C}{\epsilon c}(\sigma\sqrt{N_{tol}} + \sqrt{\sigma^2 N_{tol} + ||\mathbf{X}||_F^2}\sqrt{\sum_{i=1}^{4}(1 - \frac{r_i}{R_i})}) \tag{19}$$

In particular, when $r_i = R_i \quad i = 1, \cdots, 4$*, Equation 19 degenerates into*

$$P(||\hat{\mathbf{X}} - \mathbf{X}||_F \leq \epsilon) \geq 1 - \frac{C\sigma\sqrt{N_{tol}}}{\epsilon c} \tag{20}$$

Corollary 1 characterizes the probability that the F-norm distance between our estimated value $\hat{\mathbf{X}}$ and true value $\mathbf{X}$ is less than some small amount ϵ. When the hyperparameter $\mathbf{r}$ gets closer to rank $\mathbf{R}$, the probability will gets bigger. It means that the more accurate the selection of hyperparameters, the higher the efficiency of the algorithm. Besides this, the Corollary 1 also shows that when the the variance of the noise σ^2 decreases, the norm of the real tensor $||\mathbf{X}||_F$ or the total number of elements N_{tol} is small, probability of small estimation error will also be bigger. Corollary 1 ensures that the estimated value given by our model will be close to the true value with a high probability when the parameters are selected reasonably

Besides that, for continues case FEN model, Equation 6, we also has the following theorem. Similar to the discrete case, we set $\alpha_k = 0$.

Theorem 3. *If the* $Y_{ij}(t)$ *is square integrable function, we can get*

$$\mathbb{E}\{||\mathcal{Y} - \hat{\mathcal{X}}||_F^2\} \leq 2n^2 \sum_{k=K+1}^{\infty} \lambda_k + 4\lambda_1 nK(n - s) \tag{21}$$

where $\{\lambda_k\}$ *is a non-negative monotonically decreasing sequence and* $\sum_{k=K+1}^{\infty} \lambda_k \to 0$ *as* $K \to \infty$*.*

Theorem 3 describes the up bound of the distance between our best estimation and the true function in the continue case. The up bound consists of two part, the first part $2n^2 \sum_{k=K+1}^{\infty} \lambda_k$ is caused by functional PCA. It can also be considered as the information loss of decomposition of the third continues dimension. When K increases, the first part of the up bound will be small, even be 0 when $K \to \infty$. The second part of up bound $4\lambda_1 nK(n - s)$ is caused by the tucker decomposition of the first two discrete dimensions. When K increases, the second part of up bound will be big. Because the bigger tensor which needs to be decomposed, the more information will lose. In this way, when K increases, we will lose less information in the continues dimension decomposition but more in the first two discrete dimensions. The opposite happens when K is small. So we need to trade off the value of K to minimize the up bound in the application. Other parameters of Equation 21 also fits our perception. The lower total number of function n, the bigger target dimension number s will lead to lower up bound.

6 SIMULATION

In the simulation study, we evaluate the performance of the proposed FEN method and compare it with other baseline methods for both small-scale and large-scale functional network data with different missing percentage and variance of noise.

6.1 Introduction of the Baseline methods and Data Generation

As we have mentioned in the literature review, we mainly choose the PCA completion, dynamic network model and tensor completion these three kinds of algorithm as the baseline methods. 1) PCA algorithms: a)VPCA(Nomikos and MacGregor, 1995), b)MFPCA(Paynabar et al., 2016), c) SIFPCA(Yao et al., 2005); 2)Dynamic Network algorithms : a)MTR(Hoff, 2015), b)LDS and MLDS(Rogers et al., 2013) ; 3) Tensor Completion algorithms: a) SPC(Yokota et al., 2016) b) t-TNN(Hu et al., 2015).

Then we can compare these methods with FEN model. But first, we need to interpolate and complete the observations to regular observations for the algorithm that can not deal with irregular observations.

According Equation 3, without considering the mean function, we sequentially generate tensor core $\mathbf{B}$ which needs to satisfy $\mathbf{rank}_1(\mathbf{B}) = \mathbf{rank}_2(\mathbf{B}) = s, \mathbf{rank}_3(\mathbf{B}) = K$, discrete matrix basis Φ which needs to be column orthogonal and continues matrix basis Θ which is generated by Fourier bases defined on $[-1, 1]$ to guarantee orthogonality. After getting continue tensor $\mathcal{X}$ by Equation 3, we can set the size of observe set L to equidistant cut it on the $[T_s, T_e]$ to get $\mathbf{X}$. Then we can add the observe noise $\mathcal{E}$ with the vari-

Table 1: small-scale simulation parameter setting

Parameter	$dim(\mathcal{X})$	L	$dim(\mathbf{B})$	**r**
Value	[10, 10, [-1,1]]	50	[3,3,8]	[3, 3, 8]
Parameter	σ^2	α_k	ω	
Value	0.01,0.1,0.2	0.1	40% , 30% , 20% , 10%	

ance of observation noise σ^2. At last, we need to set the missing percentage ω. In the full observation tensor $\mathbf{Y}^F$, we randomly select the elements with ratio ω as the missing value, and at the same time, we set the mask tensor Ω as 0 at the corresponding position, 1 at the rest position. In this way, we have a set of functional network data with irregular observation for simulation experiments.

6.2 Small-Scale Simulation

The parameter setting is shown in the Table 1. For each methods and each setting of parameters, we generate data and run the algorithm 20 times and calculate the $MSE = ||\hat{\mathbf{X}} - \mathbf{X}||_F^2$. Table 2 shows the mean of MSE for each method under $\sigma^2 = 0.2$ where standard deviation of MSE are in brackets. The results under different σ^2 are in supplement.

Table 2: $MSE(\times 10^{-2})$ of different algorithms when $\sigma^2 = 0.2$ at small scale simulation

Algorithm	$\omega = 40\%$	$\omega = 30\%$	$\omega = 20\%$	$\omega = 10\%$
VPCA	1960(5740)	104(33.1)	50.4(33.0)	6.23(1.58)
MFPCA	587(495)	145(89.5)	30.8(10.5)	8.34(3.36)
SIFPCA	4420(486)	4420(470)	4390(475)	4390(475)
SPC	18.1(4.66)	9.03(2.62)	4.08(0.865)	1.42(0.248)
t-TNN	61.3(69.0)	14.9(6.38)	7.14(8.99)	1.41(0.361)
LDS	7040(6810)	5340(1280)	5220(1494)	5250(1140)
MLDS	2560(6550)	174(86.9)	103(98.0)	26.7(50.0)
MTR	2330(1270)	1770(1360)	1240(1220)	1240(1220)
FEN	3.97(0.112)	3.81(9.94e-2)	3.75(7.09e-2)	3.69(6.52e-2)

We can find that, when the noise variance σ^2 decreases or missing percentage ω decreases, MSE of each method will become small which fits our intuition. FEN model has better performance than baseline methods and is even less affected by the missing percentage. VPCA, MFPCA and SIFPCA regard different functions as a repeated sampling of one function which is not in line with the real state of the data. This makes them have bad performance. As to LDS, MLDS and MTR algorithm, Because they could not deal with the problem of irregular observations, they had to interpolate first and then estimate, which resulted in poor results. Among them, MLDS algorithm has the best performance but is also greatly affected by the missing ratio. The two tensor completion algorithms, SPC and T-TNN, have similar performance with FEN model, but the interpretability is not as good as FEN model because the dimension reduction of community structure is not considered.

6.3 Big-Scale Simulation

Similarly to the last subsection, we first give the parameter setting of the big-scale simulation in Table 3.

Table 3: big-scale simulation parameter setting

Parameter	$dim(\mathcal{X})$	L	$dim(\mathbf{B})$	**r**
Value	[50, 50, [-1,1]]	100	[15,15,25]	[15, 15, 25]
Parameter	σ^2	α_k	ω	
Value	0.01,0.1,0.2	0.1	40% , 30% , 20% , 10%	

We also calculate each parameter's mean and standard deviation of MSE of 20 times running for comparison. The specific simulation results are shown in Table Table 4. The SIFPCA algorithm is not used in the big scale simulation due to its computational complexity. The results under different σ^2 are in supplement.

Table 4: MSE of different algorithms when $\sigma^2 = 0.2$ at big scale simulation

Algorithm	$\omega = 40\%$	$\omega = 30\%$	$\omega = 20\%$	$\omega = 10\%$
VPCA	1725(964)	466(138)	170(60.6)	41.4(4.04)
MFPCA	2337(3203)	641(432)	151(29.2)	40.3(2.34)
SPC	28.8(0.724)	16.4(0.469)	8.53(0.239)	3.34(0.0931)
t-TNN	12.5(2.85)	0.298(0.189)	0.0509(0.0047)	0.015(3.71e-4)
LDS	866(265)	652(13.2)	584(7.28)	554(5.99)
MLDS	1267(732)	722(192)	568(67.0)	531(5.52)
MTR	1268(724)	1008(90.3)	736(48.6)	625(23.6)
FEN	0.0396(1.31e-4)	0.0393(7.70e-5)	0.0390(1.15e-4)	0.0388(1.03e-4)

We can find that when the network dimension and observation time point increase, the advantages of FEN model over other algorithms are further improved, and it also shows robustness to different missing percentage. The rest of the results are similar to those of small-scale simulations and will not be described here.

7 CASE STUDY

7.1 Data Introduction

In this section, we use Hong Kong subway data to compare the performance of FEN model with other baseline models to show the advantages of our method.

The Hong Kong subway network is a huge network model, the subway stations are the nodes of the network model and we use the passenger flow between two subway stations as the weight function between two nodes. At the same time, the observation time of passenger flow data between different subway stations may be different due to its scale, which satisfies the setting of irregular observation. In addition, the subway passenger flow is a function that changes over time in a day and we can regard the observation in different days as different samples. Under the above conditions, it is a perfect data to use FEN model to do the estimation and prediction.

The data parameters of the subway network in Hong Kong are shown in Table 5, in which the hyperparameter **r** is the

Table 5: Parameter Setting of Hong Kong subway

Parameter	$dim(\mathcal{X})$	L	$\mathbf{r}$
Value	[90, 90, [5:00am,12:00pm]]	99	[23,23,20]
Parameter	α_k	ω	N
Value	0.1	40% , 30% , 20% , 10%	24

Table 6: MSE of FEN and PCA methods of Hong Kong subway

ω	**SPC**	**t-TNN**	**FEN**
40%	5.30(0.283)	11.3(0.358)	4.12(0.247)
30%	4.07(0.225)	7.83(0.345)	4.63(0.236)
20%	3.79(0.197)	3.81(0.331)	3.61(0.211)
10%	3.61(0.189)	1.77(0.214)	3.59(0.179)

optimal hyperparameter selected by pre-examination. The missing data are randomly selected.

7.2 Compared with Baseline

Since the Hong Kong subway network is so huge that the dynamic network models with high computational complexity can not fit it . So we only consider the PCA methods and tensor completion methods.

Because the tensor completion methods can not handle data sampled multiple times, we also set the sample number N of FEN model. We use the data for 24 days to repeat these algorithms and compare the mean and standard deviation of $MSE_{miss} = ||\mathcal{P}_{\Omega^C}(\hat{\mathbf{X}} - \mathbf{X}^D)||_F^2$ in 24 times running. The results are shown in Table 7. The data in parentheses for the corresponding column in the table indicate the corresponding standard deviation.

Since PCA series algorithms can choose different ratio of principal components to achieve different algorithm efficiency for multiple sampled data, we need to use the FEN model with multi-sampling to do the comparison, but sampling data need to be divided into training set and test set to proper number of principal components in PCA algorithms. We need to make the MSE of PCA algorithms and FEN on the training set close, then compare the MSE of different algorithms on test set . We use MSE_{train} and MSE_{test} to denote the MSE on the training set and testing set. Similar to the PCA methods, the train model of FEN and the test model of FEN has the same matrix bases Φ and Θ, we only need to calculate the core tensor of test set, $\mathbf{B}_{test}$, as following,

$$\begin{aligned} \hat{\mathbf{B}}_{test} &= \mathbf{Y}_{test} \times_1 \hat{\Phi} \times_2 \hat{\Phi} \times_3 \hat{\Theta} \\ \hat{\mathbf{X}}_{test} &= \hat{\mathbf{B}}_{test} \times_1 \hat{\Phi} \times_2 \hat{\Phi} \times_3 \hat{\Theta} \end{aligned} \tag{22}$$

As shown in Table 6, the data in parentheses for the corresponding column represents MSE_{train}, the data outside parentheses represents MSE_{test}.

It can be seen that FEN model is better than SPC and t-TNN in most cases,but the advantage is not significant. On the one hand, there are still some differences between the real world data and the theoretical functional network model, which cannot be fully fitted. On the other hand, the selection of hyperparameter $\mathbf{r}$ may also have some influence, and the optimal hyperparameter may change under different missing percentage. Compared with PCA methods, we can find that in most cases, the FEN model can keep its MSE_{test} smaller when MSE_{train} is greater than or equal to MSE_{train} of MFPCA. However, due to its poor effect, VPCA algorithm cannot fit the data well even when its all principal components are used, so its MSE_{train} and MSE_{test} are both large. It can be seen that our FEN model still has certain advantages compared with PCA algorithms.

Table 7: MSE of FEN and tensor completion methods of Hong Kong subway

ω	**VPCA**	**MFPCA**	**FEN**
40%	23.0(22.7)	12.0(2.63)	13.0(4.20)
30%	22.8(22.5)	11.8(2.62)	10.9(5.73)
20%	22.8(22.5)	11.7(2.62)	8.99(6.57)
10%	22.8(22.5)	11.7(2.62)	7.79(7.59)

8 CONCLUSION

We propose a method of modeling functional network data, FEN model. It can solve the function smoothness constraint problem, improve the interpretability by using the community structure to reduce dimension and deal with irregular observation. Then the conjugate gradient method under Riemann optimization is used to estimate the model. In terms of theoretical properties, three theorems and an important corollary are proposed for the FEN model to ensure that asymptotic convergence of its solvability and solution. We compare the FEN model with other models both on the simulation data and Hong Kong subway data to high lighte the advantages of our algorithm in completing functional network data.

However, there are still some problems in this model, which need further research and improve. 1) Assumption 2 may be a little strong so we should try to prove it in future work to make the theoretical properties more rigorous; 2) The core tensor of the test set applied to the real case is not accurately calculated by Equation 22, which leads to poor performance of the test set when the proportion of missing observations is large.

References

Bali, J. L., Boente, G., Tyler, D. E., and Wang, J.-L. (2011). Robust functional principal components: A projection-pursuit approach. *The Annals of Statistics*, 39(6):2852–2882.

Batselier, K., Chen, Z., and Wong, N. (2017). Tensor network alternating linear scheme for mimo volterra system identification. *Automatica*, 84:26–35.

Chun, H., Chen, M., Li, B., and Zhao, H. (2013). Joint conditional gaussian graphical models with multiple sources of genomic data. *Frontiers in genetics*, 4:294.

Danaher, P., Wang, P., and Witten, D. M. (2014). The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(2):373–397.

De Lathauwer, L., De Moor, B., and Vandewalle, J. (2000). A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278.

Durante, D., Mukherjee, N., and Steorts, R. C. (2017). Bayesian learning of dynamic multilayer networks.

Foutz, N. Z. and Jank, W. (2010). Research note—prerelease demand forecasting for motion pictures using functional shape analysis of virtual stock markets. *Marketing Science*, 29(3):568–579.

Goulart, J. H. d. M., Boizard, M., Boyer, R., Favier, G., and Comon, P. (2015). Tensor cp decomposition with structured factor matrices: Algorithms and performance. *IEEE Journal of Selected Topics in Signal Processing*, 10(4):757–769.

Greven, S., Crainiceanu, C., Caffo, B., and Reich, D. (2011). Longitudinal functional principal component analysis. In *Recent Advances in Functional Data Analysis and Related Topics*, pages 149–154. Springer.

Hoff, P. D. (2015). Multilinear tensor regression for longitudinal relational data. *The annals of applied statistics*, 9(3):1169.

Hu, W., Tao, D., Zhang, W., Xie, Y., and Yang, Y. (2015). A new low-rank tensor model for video completion. *arXiv preprint arXiv:1509.02027*.

Kolar, M. and Xing, E. (2011). On time varying undirected graphs. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 407–415. JMLR Workshop and Conference Proceedings.

Kolda, T. G. and Bader, B. W. (2009). Tensor decompositions and applications. *SIAM review*, 51(3):455–500.

Liu, J., Zhu, C., Long, Z., Huang, H., and Liu, Y. (2021). Low-rank tensor ring learning for multi-linear regression. *Pattern Recognition*, 113:107753.

Loyal, J. D. and Chen, Y. (2021). An eigenmodel for dynamic multilayer networks. *arXiv preprint arXiv:2103.12831*.

McNeil, M. J., Zhang, L., and Bogdanov, P. (2021). Temporal graph signal decomposition. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1191–1201.

Nomikos, P. and MacGregor, J. F. (1995). Multivariate spc charts for monitoring batch processes. *Technometrics*, 37(1):41–59.

Oseledets, I. V. (2011). Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317.

Paynabar, K., Zou, C., and Qiu, P. (2016). A change-point approach for phase-i analysis in multivariate profile monitoring and diagnosis. *Technometrics*, 58(2):191–204.

Qiao, X., Guo, S., and James, G. M. (2019). Functional graphical models. *Journal of the American Statistical Association*, 114(525):211–222.

Qiao, X., Qian, C., James, G. M., and Guo, S. (2020). Doubly functional graphical models in high dimensions. *Biometrika*, 107(2):415–431.

Robinson, L. F. and Priebe, C. E. (2012). Detecting time-dependent structure in network data via a new class of latent process models. *arXiv preprint arXiv:1212.3587*.

Rogers, M., Li, L., and Russell, S. J. (2013). Multilinear dynamical systems for tensor time series. *Advances in Neural Information Processing Systems*, 26.

Sewell, D. K. and Chen, Y. (2017). Latent space approaches to community detection in dynamic networks. *Bayesian analysis*, 12(2):351–377.

Xu, Q., Mao, B., and Bai, Y. (2016). Network structure of subway passenger flows. *Journal of Statistical Mechanics: Theory and Experiment*, 2016(3):033404.

Yao, F., Müller, H.-G., and Wang, J.-L. (2005). Functional data analysis for sparse longitudinal data. *Journal of the American statistical association*, 100(470):577–590.

Yokota, T., Zhao, Q., and Cichocki, A. (2016). Smooth parafac decomposition for tensor completion. *IEEE Transactions on Signal Processing*, 64(20):5423–5436.

Zhong, C., Manley, E., Arisona, S. M., Batty, M., and Schmitt, G. (2015). Measuring variability of mobility patterns from multi-day smart-card data. *Journal of Computational Science*, 9:125–130.

Zhou, S., Lafferty, J., and Wasserman, L. (2010). Time varying undirected graphs. *Machine Learning*, 80(2):295–319.

Zhu, H., Strawn, N., and Dunson, D. B. (2016). Bayesian graphical models for multivariate functional data.