Bayesian Kernelized Matrix Factorization for Spatiotemporal Traffic Data Imputation and Kriging

Mengying Lei, Aurelie Labbe, Yuankai Wu, and Lijun Sun, *Member, IEEE*

***Abstract*— Missingness and corruption are common problems for real-world traffic data. How to accurately perform imputation and prediction based on incomplete or even sparse traffic data becomes a critical research question in intelligent transportation systems. Low-rank matrix factorization (MF) is a common solution for the general missing value imputation problem. To better characterize and encode the strong spatial and temporal consistency in traffic data, existing work has introduced flexible spatial/temporal Gaussian process (GP) priors to model the latent factors in MF framework, which also allows us to perform *kriging* for unseen locations and virtual sensors. However, learning the hyperparameters in GP kernels remains a challenging task. In this paper, we present a Bayesian kernelized matrix factorization (BKMF) model with an efficient Markov chain Monte Carlo (MCMC) sampling algorithm for model inference. By learning the kernel hyperparameters from their marginal posteriors through a slice sampling treatment and updating the latent factors alternatively with Gibbs sampling, we achieve a fully Bayesian model for the spatiotemporally kernelized (i.e., GP prior regularized) MF framework. We apply BKMF on both imputation and kriging tasks, and our results demonstrate the superiority of BKMF compared with state-of-the-art spatiotemporal models. In addition, we also explore the effects of different GP kernels in characterizing networked spatiotemporal traffic state data.**

***Index Terms*— Missing data imputation, kriging, Gaussian process, low-rank matrix factorization, Markov chain Monte Carlo.**

I. Introduction

TRAFFIC state (e.g., speed/flow) is a typical type of multivariate data that evolves over both space (e.g., on a transportation network) and time [1]. Learning the spatiotemporal patterns from traffic data is essential in transportation system analysis [2]. One of the most fundamental and common issues in traffic sensing data is the "missingness" problem due to various factors such as sensor failure/malfunctions or network communication errors. In addition, traffic state data collected from emerging crowdsourcing and moving sensing systems (e.g., Google Waze) is often naturally sparse due to the limited spatial and temporal coverage. The missingness problem makes it difficult to use the data directly for downstream applications that often require complete information, such as traffic forecasting, route navigation, and travel time estimation. Therefore, imputing missing values has been a critical task in spatiotemporal traffic data analysis.

Traffic data collected from a sensing network can naturally be organized into a multivariate time series matrix, with each row representing a time series collected from a sensor. In general, there are two types of "missing" patterns. The first is random missing, where the time series observed from a particular sensor contains missing values. This case of missingness is often caused by sensor and communication errors in the system. The second corresponds to the scenario where no sensors are installed at a particular location (i.e., whole row missing), but readings from nearby sensors are available for us to perform interpolation for the virtual sensors. We refer to the estimation of these two types of missing values as *imputation* and *kriging*, respectively [3], [4]. Given the inherent global patterns observed in spatiotemporal data, latent factor-based low-rank matrix factorization (MF) methods have been widely used for missing data estimation [5]. However, the standard MF framework [6] is invariant to row and column permutation; thus, it does not account for the spatial and temporal correlations in the data, which are essential for data completion. To incorporate such local dependency, various spatial or temporal regularized MF models have been proposed. For instance, Rao *et al.* [7], Wang *et al.* [8], Bahadori *et al.* [3] imposed spatial consistency using a graph Laplacian regularization, Xiong *et al.* [9] modeled temporal consistency using a local Markov structure, and Yu *et al.* [5] captured temporal dependence through an autoregressive (AR) regularizer. Although these regularization methods can encode spatiotemporal correlations, one should note that the definition of spatial and temporal correlation structure also plays an ultimate role in model performance. For instance, most previous studies (e.g., [3], [10]) use a square exponential function $\exp\left(-\text{dist}(i, j)^2/l^2\right)$ to define link weight in an adjacency matrix with a pre-specified lengthscale hyperparameter l; however, the lengthscale l is critical in characterizing the structure of spatial process, and should be carefully learned or specified using extensive cross-validation. Similarly, in terms of the temporal structure, a simple AR model is hardly able

Manuscript received July 27, 2021; revised December 5, 2021 and February 1, 2022; accepted March 20, 2022. This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada, in part by the Fonds de recherche du Québec—Nature et technologies (FRQNT), and in part by the Canada Foundation for Innovation (CFI). The work of Mengying Lei and Yuankai Wu was supported by the Institute for Data Valorization (IVADO). The Associate Editor for this article was L. Li. *(Corresponding author: Lijun Sun.)*

Mengying Lei, Yuankai Wu, and Lijun Sun are with the Department of Civil Engineering, McGill University, Montreal, QC H3A 0C3, Canada (e-mail: mengying.lei@mail.mcgill.ca; yuankai.wu@mail.mcgill.ca; lijun.sun@mcgill.ca).

Aurelie Labbe is with the Department of Decision Sciences, HEC Montréal, Montréal, QC H3T 2A7, Canada (e-mail: aurelie.labbe@hec.ca).

Digital Object Identifier 10.1109/TITS.2022.3161792

to capture both short-term (e.g., hourly) and long-term (e.g., daily/weekly) temporal correlations [11] of the data.

In spatial statistics and machine learning, Gaussian process (GP) is a widely used tool to model spatiotemporal data [12], [13]. The key idea of GP is to capture the complex covariance between any pair of observations using a simple kernel structure with only a few hyperparameters. In addition, the probabilistic nature of GP also allows us to quantify the uncertainty in the data. However, the main challenge associated with GP models is the high computational cost ($\mathcal{O}(n^3)$ mainly because of the Cholesky decomposition of the covariance matrix, where n is the sample size). In our case, learning a GP for a $M \times N$ data matrix with no missing values has a complexity of $\mathcal{O}(M^3N^3)$. To address the computational issue, Luttinen and Ilin [14] developed a Gaussian process factor analysis (GPFA) model by combining GP with MF in a single probabilistic model, in which separate GP priors are introduced to model the spatial and temporal factors, respectively. With the low-rank factorization scheme and separate kernels, GPFA reduces the computational cost to $\mathcal{O}(M^3D^3 + N^3D^3)$ when updating the factor matrices as a whole, or to $\mathcal{O}(M^3D + N^3D)$ if columns of the latent factors are updated separately, where $D \ll (M, N)$ is a predefined rank for MF (i.e., size of the latent dimension).

Despite the efficient structure, one main challenge with GPFA is that the hyperparameters of the GP kernels — which play a crucial role in model interpretation and performance — are difficult to estimate. In GPFA [14], the authors proposed to use variational Bayes approximation to infer the joint posterior over latent factors. Based on these approximations, kernel hyperparameters are estimated by maximizing the lower bound of the likelihood of the corresponding factors. However, this strategy assumes that the two factor matrices are independent, which may decrease the estimation accuracy. More importantly, since the kernel hyperparameters are estimated conditioning on the latent factors, the strong relationship between them is essentially ignored [15]. Such estimation in a hierarchical GP model usually generates an extremely narrow posterior distribution for the hyperparameters [16], [17], which makes hyperparameter learning challenging.

In this paper, we present a GP regularized Bayesian Kernelized Matrix Factorization (BKMF) framework by adapting the GPFA method developed in [14] to complete missing values in spatiotemporal traffic data. To address the kernel hyperparameter learning problem in GPFA, we compute the marginal posteriors by integrating out the latent factors and achieve the MCMC updates via slice sampling [18]. The fully Bayesian framework allows us to exploit the spatiotemporal correlations without prior knowledge. We show that our proposed BKMF model can be effectively used for spatiotemporal data imputation and kriging, even in the presence of large amounts of missing values. Furthermore, the interpretable kernels in BKMF provide a clear representation for real-world spatiotemporal traffic patterns. The overall contribution of this work is threefold:

1) We introduce GP kernel regularization into the MF framework to model spatiotemporal traffic data. By placing flexible kernel priors over all columns of the latent matrices properly, the spatial and temporal dependencies among the data rows and columns can be well captured. The proposed model can perform imputation for extremely sparse data (e.g., 95% data is missing) and achieve complex spatiotemporal kriging.
2) We develop an MCMC-based fully Bayesian sampling method for model inference, which is free from parameter tuning. Through Gibbs sampling and slice sampling, both the model parameters and kernel hyperparameters can be efficiently updated. The Bayesian framework also allows us to quantify the uncertainty of the estimation.
3) We conduct extensive experiments in terms of both imputation and kriging tasks on two traffic data sets, where the effectiveness of various GP kernels is also explored. The completion results show the superiority of BKMF over existing low-rank spatiotemporal analysis models especially in the kriging scenarios.

The remainder of this paper is organized as follows. Sec. II reviews related studies for spatiotemporal data imputation and kriging, and also presents the key research challenge. Sec. III illustrates the tasks/problems of this work and introduces the formulation that incorporates spatial and temporal side information into low-rank matrix factorization. Sec. IV presents the proposed BKMF model and the MCMC algorithm for model inference. In Sec. V, we apply BKMF on two real-world traffic speed data sets for both imputation and kriging tasks and also examine the effects of choosing different kernel functions. Sec. VI summarizes this study and discusses some model properties and future research.

II. Related Work

Missing value imputation and kriging for spatiotemporal traffic data are the two tasks discussed in this paper. Traditionally, the kriging tasks are commonly addressed by GP-based models. The low-rank structured models have also been widely explored for spatiotemporal data completion problems.

A. Gaussian Process (GP) for Spatiotemporal Data Kriging

GP is a common approach to analyze spatiotemporal data, particularly for the kriging tasks. Kriging estimates the time series/signals of new locations by taking advantage of the spatial correlations among the observed locations [19], where the spatial dependency structure is modeled through a specific covariance or variogram function [20]. In general, the data are decomposed into a trend component and two residual components—a spatial residual a GP with zero means and the a pure error term (i.e., nugget effect). Depending on the assumptions of the trend part, different types of kriging methods are deduced, including universal kriging, ordinary kriging, etc. These kriging models have been applied for traffic data analysis, for example, to estimate annual average daily traffic for unobserved locations [21]. However, kernel functions need to be specified in classical kriging models, and the kernel hyperparameter estimation is computationally expensive. Recent research also extends spatial kriging to spatiotemporal kriging, accounting for both spatial and temporal correlations, such as in [22], [23]. However, these

TABLE I
LOW-RANK MODELS FOR SPATIOTEMPORAL DATA ANALYSIS

Methods	Model structure	Training algorithm
[7]	graph Laplacian, MF	MAP
[3], [24]	graph, VAR, tensor learning	MAP
[8]	graph, one-order dynamic model, MC	convex optimization
[5]	AR, MF	MAP
[9]	one-order dynamic model, TF	MCMC
[25]	AR, MF/TF	MCMC
[26]	AR, TC	convex optimization
[27]	GP, MF	MAP
[14]	GP, MF	variational inference

MAP: maximum a posteriori; VAR: vector AR;
MC: matrix completion; TC: tensor completion; TF: tensor factorization.

studies generally extended the idea of variogram/covariance modeling, which still needs domain knowledge and manual effort. Although GP offers a framework for characterizing complex spatiotemporal correlations, the main bottleneck for real-world applications is the high computational cost.

B. Low-Rank Models for Missing Data Imputation and Kriging

Low-rank factor models provide an alternative dimensionality reduction framework for missing data imputation of large-scale spatiotemporal data sets. In default MF, the imputation result is invariant to the permutation of rows and columns of the data matrix, which implies that the strong spatial and temporal patterns are ignored. Therefore, the main idea of low-rank spatiotemporal models is to introduce regularization terms to account for spatial/temporal structure in addition to the standard low-rank framework. In Table I, we briefly review some existing models in the literature.

Most studies rely on graph-based regularization to encode local spatial and temporal consistencies. Despite the simplicity, there are several issues with these models. First, tuning the regularization parameters takes a lot of computing time. Second, most studies define the edge weights in graph regularization using a specific kernel structure; however, since the hyperparameters (such as lengthscale and variance in an exponential kernel) are essential to the underlying processes, they need to be carefully tuned instead of being assigned empirically. Third, real-world spatial and temporal processes often cannot be fully characterized using a graph structure. This is, for example, the case with long-term periodic correlations and spatial connections described with geographical distances. AR modeling is another popularly applied regularization way. However, the AR regularized models that consider only temporal characteristics cannot account for spatial patterns and therefore are not able to deal with the kriging tasks. In addition to the GP and low-rank structured completion methods, there also exist missing data completion approaches using neural networks [4] or other types of data such as video camera records and GPS trajectory data [28] for traffic analysis. However, these approaches usually require expensive computational resources.

It should be noted that in citywide traffic state estimation, the kriging task becomes highly important and challenging. Given the high installation and maintenance costs of traditional fixed sensing techniques (e.g., loop detectors or surveillance cameras), the sensor detectors usually only cover a small fraction of the road network in the whole city, leaving large chunks of roads without sensors (whole row missing) of which the data needed to be estimated. In this case, it is important to incorporate a proper spatial process in order to leverage the data from available sensors to estimate/interpolate variables at unseen locations. Conceptually, the kriging task in traffic data analysis is equivalent to installing "virtual sensors" in a network. Some recent studies have proposed to incorporate correlations learned from multiple data sources, e.g., fusing camera data and crowdsourcing floating car data [29], or combining limited sensor data with additional simulation modules [30] to achieve kriging in citywide traffic inference.

III. PRELIMINARIES

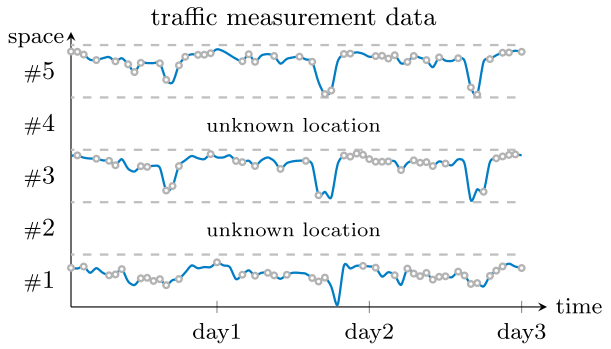
A. Notations

Throughout this paper, we use lowercase letters to denote scalars (e.g., x), boldface lowercase letters to denote vectors (e.g., $\boldsymbol{x} \in \mathbb{R}^M$), and boldface capital letters to denote matrices (e.g., $\boldsymbol{X} \in \mathbb{R}^{M\times N}$). For a matrix $\boldsymbol{X} \in \mathbb{R}^{M\times N}$, we denote the ith row and jth column by $\boldsymbol{X}_{i:}$ and $\boldsymbol{X}_{:j}$, respectively, and the elements are denoted by $x_{i,j}$ or $[\boldsymbol{X}]_{ij}$. The Frobenius norm of $\boldsymbol{X}$ is defined by $\|\boldsymbol{X}\|_F := \sqrt{\sum_{m=1}^{M}\sum_{n=1}^{N} x_{m,n}^2}$. The transpose and trace of $\boldsymbol{X}$ are denoted by $\boldsymbol{X}^\top$ and $tr(\boldsymbol{X})$, respectively. We define the vectorization of $\boldsymbol{X}$ as its column concatenation (i.e., by stacking the column vectors of $\boldsymbol{X}$), and denote it by $\boldsymbol{X}_{:}$ or $\text{vec}(\boldsymbol{X}) \in \mathbb{R}^{MN}$. The operations $\boldsymbol{A} \otimes \boldsymbol{B}$ and $\boldsymbol{A} \odot \boldsymbol{B}$ denote the Kronecker product and element-wise multiplication of matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, respectively. Finally, $\boldsymbol{I}_M$ denotes an $M \times M$ identity matrix and $\boldsymbol{1}_M$ denotes a length-M column vector of ones.

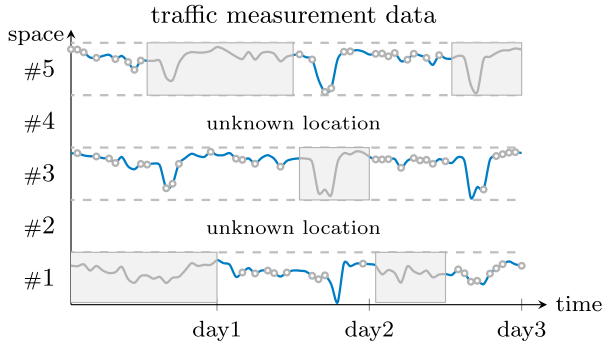
B. Problem Description

In this work, we focus on two tasks: imputation and kriging for spatiotemporal data, given a high ratio of missing values. We consider here a partially observed data matrix $\boldsymbol{Y} \in \mathbb{R}^{M\times N}$, where rows correspond to sensors at M locations and columns correspond to N time points that are uniformly distributed. The elements of $\boldsymbol{Y}$ represent traffic state variables, e.g., traffic speed or flow at a specific location and time. Observed data points are denoted by $\mathcal{P}_\Omega(\boldsymbol{Y})$, in which Ω refers to the set of observed indices and $\mathcal{P}_\Omega(\cdot)$ is the projection operator retaining entries with indices in Ω. We consider a high ratio ($> 50\%$) of missing data, and the goal is to complete the missing values (with indices in Ω^c) based on the observed data $\mathcal{P}_\Omega(\boldsymbol{Y})$ in different missing scenarios.

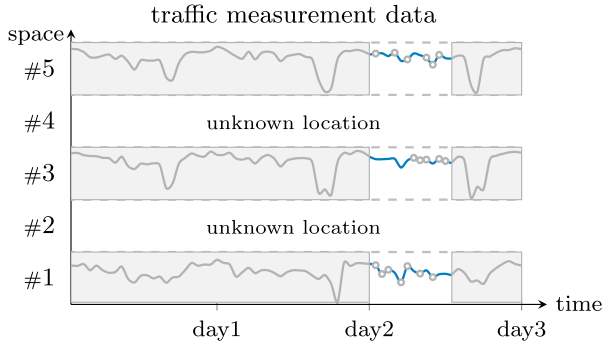
We refer to kriging as a special imputation task for certain rows in $\boldsymbol{Y}$ that are completely missing. For instance, in the case of Figure 1 with five sensors, kriging refers to the estimation of signals at unknown sensors $\{\#2, \#4\}$ based on the incomplete observations from sensors $\{\#1, \#3, \#5\}$ by leveraging the spatial dependencies among the five sensors. Specifically, for the kriging tasks, we consider three types of missing scenarios as illustrated in Figure 1: (a) random missing (RM), i.e., data at sampled locations are missing randomly; (b) random block



(a) Kriging with random missing (K-RM).



(b) Kriging with random block missing (K-RB).



(c) Kriging with nonrandom block missing (K-NB).

Fig. 1. Illustration of the kriging tasks in three missing scenarios. Observed data are represented in blue lines, random missing points and consecutive missing points are represented in gray circles and gray lines, respectively.

missing (RB), where observations at the sampled locations are corrupted during a time interval; (c) nonrandom block missing (NB), where data are missing at some consecutive timestamps for all sampled sensors. Both random and nonrandom block missing in (b) and (c) also include random missing values. These missing scenarios correspond to time-varying sensor availability in real-world applications: some sensors might be corrupted/not functional at random or continuous-time points, while new sensors need to be introduced. Note that when performing these tasks, we have additional spatial information (e.g., network typology, sensor location) for the row indices in matrix $\boldsymbol{Y}$ to encode spatial consistency in the model.

C. Classic MF Models

Given an incomplete $M \times N$ data matrix $\boldsymbol{Y}$, standard MF decomposes the matrix as a product of two low-dimensional latent factor matrices $\boldsymbol{U} \in \mathbb{R}^{M\times D}$ and $\boldsymbol{V} \in \mathbb{R}^{N\times D}$ ($D \ll \{M, N\}$):

$$\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{V}^\top + \text{noise}. \tag{1}$$

Training such a model involves finding the best rank-D approximation to the partially observed data matrix given a specified loss function. With a probabilistic framework, the Probabilistic Matrix Factorization (PMF) model imposes a zero-mean Gaussian assumption on each row of the latent factors and estimates parameters using Maximum a Posteriori (MAP) estimation [6]. The objective function is:

$$\min_{\boldsymbol{U},\boldsymbol{V}} \frac{1}{2}\left\|\mathcal{P}_\Omega(\boldsymbol{Y}-\boldsymbol{U}\boldsymbol{V}^\top)\right\|_F^2 + \frac{\lambda_U}{2}\|\boldsymbol{U}\|_F^2 + \frac{\lambda_V}{2}\|\boldsymbol{V}\|_F^2, \tag{2}$$

where λ_U and λ_V are regularization parameters depending on the variances of the Gaussian priors. Bayesian Probabilistic Matrix Factorization (BPMF) model [31] is further proposed as a fully Bayesian treatment of PMF to address the parameter tuning problem.

However, standard MF cannot describe the correlation among the rows/columns of the data—row/column permutation does not affect the estimation accuracy. In order to account for spatiotemporal correlations, a potential strategy is to use a graph regularization in the low-rank model. For example, Rao *et al.* [7] introduced an efficient Graph-Structured MF (GSMF) model when side information (e.g., a graph) is available. The optimization problem can be written as:

$$\begin{aligned}\min_{\boldsymbol{U},\boldsymbol{V}} \frac{1}{2}\left\|\mathcal{P}_\Omega(\boldsymbol{Y}-\boldsymbol{U}\boldsymbol{V}^\top)\right\|_F^2 + \frac{\lambda_U}{2}\|\boldsymbol{U}\|_F^2 + \frac{\lambda_V}{2}\|\boldsymbol{V}\|_F^2 \\ + \frac{\lambda_L}{2}\left\{tr\left(\boldsymbol{U}^\top\mathcal{L}_U\boldsymbol{U}\right) + tr\left(\boldsymbol{V}^\top\mathcal{L}_V\boldsymbol{V}\right)\right\},\end{aligned} \tag{3}$$

where $\mathcal{L}_U$ and $\mathcal{L}_V$ are Laplacian matrices constructed from the graphs for rows and columns, respectively. While this approach captures the row and column dependencies through the Laplacian regularizer, the optimization model also involves more weight parameters that have to be tuned carefully.

D. MF With GP Priors

A more generalized approach is to specify a covariance structure instead of a graph to encode the spatial and temporal dependencies. GPFA [14] (or Kernelized Probabilistic Matrix Factorization (KPMF) [27]) replaces the graph Laplacian regularization by introducing row-wise and column-wise kernel (covariance) functions. The model assumes that each column of the latent factors, i.e., $\boldsymbol{U}_{:d}$ and $\boldsymbol{V}_{:d}$, follows a zero mean GP prior. KPMF imposes a strong assumption that the covariance functions of priors are known in advance, while GPFA learns kernel hyperparameters. In KPMF, the latent matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ are estimated through MAP based on gradient descent. The resulting objective function is given by:

$$\begin{aligned}\min_{\boldsymbol{U},\boldsymbol{V}} \frac{1}{2\sigma^2}\left\|\mathcal{P}_\Omega(\boldsymbol{Y}-\boldsymbol{U}\boldsymbol{V}^\top)\right\|_F^2 + \frac{1}{2}\sum_{d=1}^{D}\boldsymbol{U}_{:d}^\top\boldsymbol{K}_u^{-1}\boldsymbol{U}_{:d} \\ + \frac{1}{2}\sum_{d=1}^{D}\boldsymbol{V}_{:d}^\top\boldsymbol{K}_v^{-1}\boldsymbol{V}_{:d},\end{aligned} \tag{4}$$

where $\boldsymbol{K}_u \in \mathbb{R}^{M\times M}$ and $\boldsymbol{K}_v \in \mathbb{R}^{N\times N}$ are the covariance matrices for $\boldsymbol{U}$ and $\boldsymbol{V}$, respectively.

Note that GSMF can be considered a special case of GPFA/KPMF where a regularized Laplacian kernel [32] is used as the GP prior. The main difficulty of this kernelized model is to determine the hyperparameters of the covariance/kernels. Luttinen and Ilin [14] proposed a variational Bayesian approach to estimate the model parameters and hyperparameters. However, this approximation scheme might produce inaccurate results because of the deterministic simplified estimation for the joint posteriors [31] and the neglect of the coupling between kernel hyperparameters and latent factors [17], [33]. Next, we present a complete Bayesian treatment of KPMF using Markov chain Monte Carlo (MCMC), in which both the GP hyperparameters and the latent factors can be sampled efficiently.

IV. Methodology

Following the main framework of GPFA, we introduce a new MCMC algorithm for model inference, and we call this model Bayesian Kernelized Matrix Factorization (BKMF).

A. Model Description

In keeping with the MF framework in Eq. (1), we assume an independent noise term for each row of the data, i.e., define a spatially varying noise variable $\boldsymbol{\epsilon}_m^\top \in \mathbb{R}^N, m \in \{1, \cdots, M\}$, each following a Gaussian distribution $\mathcal{N}(0, \tau_m^{-1})$. The distribution for the entries of the $\boldsymbol{Y}$ data matrix becomes:

$$y_{m,n} \sim \mathcal{N}\left(\boldsymbol{U}_{m:}\boldsymbol{V}_{n:}^\top, \tau_m^{-1}\right). \tag{5}$$

This row varying assumption for noise term considers the heterogeneity of the data, which could be more accurate in modeling real-world datasets. Note that if the number of observations in each row is too small to estimate a row-specific variance (as it is in a kriging task, for example), one can also set $\tau_m = \tau$ to define an isotropic noise distribution.

Following GPFA, we use a zero-mean GP prior to model each column of $\boldsymbol{U}$ and $\boldsymbol{V}$:

$$\begin{aligned}\boldsymbol{U}_{:d} &\sim \mathcal{GP}\left(\boldsymbol{0}, \boldsymbol{K}_d^u\right), \quad d = 1, \cdots, D,\\ \boldsymbol{V}_{:d} &\sim \mathcal{GP}\left(\boldsymbol{0}, \boldsymbol{K}_d^v\right), \quad d = 1, \cdots, D,\end{aligned} \tag{6}$$

where $\boldsymbol{K}_d^u$ and $\boldsymbol{K}_d^v$ are computed using two kernel functions $k_d^u(\cdot,\cdot;\boldsymbol{\theta}_d)$ and $k_d^v(\cdot,\cdot;\boldsymbol{\vartheta}_d)$ with hyperparameters $\boldsymbol{\theta}_d$ and $\boldsymbol{\vartheta}_d$, respectively. We ensure hyperparameters are positive by doing a reparameterization on the logarithm scale. The factor columns are assumed to be independent, thus the prior probabilities of the whole factor matrices are:

$$\begin{aligned}p\left(\boldsymbol{U}\right) &= \prod_{d=1}^{D}\mathcal{N}\left(\boldsymbol{U}_{:d} \mid \boldsymbol{0}, \boldsymbol{K}_d^u\right),\\ p\left(\boldsymbol{V}\right) &= \prod_{d=1}^{D}\mathcal{N}\left(\boldsymbol{V}_{:d} \mid \boldsymbol{0}, \boldsymbol{K}_d^v\right).\end{aligned} \tag{7}$$

We show the graphical model BKMF in Figure 2. Since there exist no conjugate priors for the GP covariance hyperparameters, we use a Gaussian prior on each log-transformed

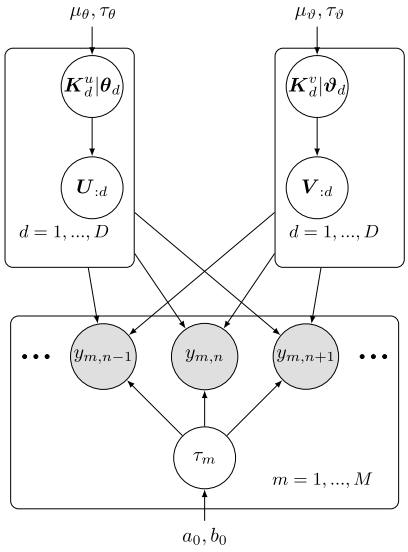


Fig. 2. The graphical model for BKMF.

kernel hyperparameter; and for each precision term τ_m, we use a conjugate Gamma prior:

$$\begin{aligned}\log\left(\theta\right) &\sim \mathcal{N}\left(\mu_\theta, \tau_\theta^{-1}\right), \quad \theta \in \{\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_D\},\\ \log\left(\vartheta\right) &\sim \mathcal{N}\left(\mu_\vartheta, \tau_\vartheta^{-1}\right), \quad \vartheta \in \{\boldsymbol{\vartheta}_1, \cdots, \boldsymbol{\vartheta}_D\},\\ \tau_m &\sim \text{Gamma}\left(a_0, b_0\right), \quad m = 1, \cdots, M.\end{aligned} \tag{8}$$

The model hyper-priors $\Theta_0 = \{\mu_\theta, \tau_\theta, \mu_\vartheta, \tau_\vartheta, a_0, b_0\}$ are predefined in the sampling process, and they should embody the prior knowledge about a specific application. Nevertheless, the selection of these initial values has little effect on the final results when the number of update iterations is sufficient, as observed in many empirical Bayesian estimation schemes.

B. MCMC-Based Prediction

In a Bayesian framework, the posterior distribution of a missing point $y_{m,n}^* : (m,n) \in \Omega^c$ is obtained by integrating out both the model parameters and hyperparameters:

$$\begin{aligned}&p\left(y_{m,n}^* \mid \mathcal{P}_\Omega(\boldsymbol{Y}), \Theta_0\right)\\ &= \int\int p\left(y_{m,n}^* \mid \boldsymbol{U}, \boldsymbol{V}, \tau_m\right)\\ &\quad \times p\left(\boldsymbol{U}, \boldsymbol{V}, \tau_m \mid \boldsymbol{\theta}, \boldsymbol{\vartheta}, a_0, b_0, \mathcal{P}_\Omega(\boldsymbol{Y})\right)\\ &\quad \times p\left(\boldsymbol{\theta}, \boldsymbol{\vartheta} \mid \mu_\theta, \tau_\theta\right) d\left\{\boldsymbol{U}, \boldsymbol{V}, \tau_m\right\} d\left\{\boldsymbol{\theta}, \boldsymbol{\vartheta}\right\},\end{aligned} \tag{9}$$

where for brevity we omit the subscript in kernel hyperparameters and use $\boldsymbol{\theta}$ and $\boldsymbol{\vartheta}$ to denote $\{\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_D\}$ and $\{\boldsymbol{\vartheta}_1, \cdots, \boldsymbol{\vartheta}_D\}$, respectively. However, the exact computation of this posterior distribution is analytically intractable because of the complex integration. With MCMC, we can approximate the posterior as:

$$p\left(y_{m,n}^* \mid \mathcal{P}_\Omega(\boldsymbol{Y}), \Theta_0\right) \approx \frac{1}{K}\sum_{k=1}^{K} p\left(y_{m,n}^* \;\middle|\; \boldsymbol{U}^{(k)}, \boldsymbol{V}^{(k)}, \tau_m^{(k)}\right), \tag{10}$$

where K is the number of samples and $\boldsymbol{U}^{(k)}, \boldsymbol{V}^{(k)}, \tau_m^{(k)}$ denote the kth sample obtained from the Markov chain with stationary distribution being the posterior over parameters and hyperparameters $\{\boldsymbol{U}, \boldsymbol{V}, \tau_m, \boldsymbol{\theta}, \boldsymbol{\vartheta}\}$.

C. Inference Process

There are two main challenges for model inference. First, there is a strong correlation between the model parameters and the hyperparameters [16]: conditioning on $\{\boldsymbol{U}, \boldsymbol{V}\}$ can induce an extremely narrow posterior over $\{\boldsymbol{\theta}, \boldsymbol{\vartheta}\}$. As a result, the Markov chain will show poor mixing. Second, the conditional distributions of GP hyperparameters are not known, implying that the Gibbs sampler is not applicable in this context.

To cope with these issues, we update the hyperparameters $\{\boldsymbol{\theta}, \boldsymbol{\vartheta}\}$ from their marginal posterior distributions with a slice sampler [18], and combine it alternately with Gibbs sampling for the model parameters $\{\boldsymbol{U}, \boldsymbol{V}, \tau_m\}$.

We introduce below each inference module specifically.

1) Sampling Factor Matrices $\{\boldsymbol{U}, \boldsymbol{V}\}$: For factor matrices $\{\boldsymbol{U}, \boldsymbol{V}\}$, the prior and likelihood are both Gaussian distributions; thus, the conditional posteriors are also Gaussian. Given that $\boldsymbol{U}\boldsymbol{V}^\top = \boldsymbol{I}_M\boldsymbol{U}\boldsymbol{V}^\top$ and $\text{vec}(\boldsymbol{AXB}) = (\boldsymbol{B}^\top \otimes \boldsymbol{A})\text{vec}(\boldsymbol{X})$ for any matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{X}$, Eq. (1) can be written as:

$$\boldsymbol{Y}_{:} = (\boldsymbol{V} \otimes \boldsymbol{I}_M)\,\boldsymbol{U}_{:} + \boldsymbol{E}_{:}, \tag{11}$$

where $\boldsymbol{E} = [\boldsymbol{\epsilon}_1; \boldsymbol{\epsilon}_2; \cdots; \boldsymbol{\epsilon}_M] \in \mathbb{R}^{M\times N}$ is the noise matrix. When $\tau_m = \tau$, the above equation can be seen as a linear model where $\boldsymbol{V} \otimes \boldsymbol{I}_M$ is the design matrix and $\boldsymbol{U}_{:}$ is the slope coefficient to be estimated. Since each column in $\boldsymbol{U}$ follows a Gaussian distribution, we can get the analytical posterior:

$$p\left(\boldsymbol{U}_{:} \mid \mathcal{P}_\Omega(\boldsymbol{Y}), \boldsymbol{V}, \tau, \boldsymbol{\theta}\right) \sim \mathcal{N}\left(\boldsymbol{U}_{:} \,\middle|\, \boldsymbol{\mu}_U^*, (\boldsymbol{\Lambda}_U^*)^{-1}\right), \tag{12}$$

where

$$\begin{aligned}
\boldsymbol{\Lambda}_U^* &= \tau\boldsymbol{H}^\top\boldsymbol{H} + \boldsymbol{K}_U^{-1},\\
\boldsymbol{\mu}_U^* &= \tau\left(\boldsymbol{\Lambda}_U^*\right)^{-1}\boldsymbol{H}^\top(\boldsymbol{Q}_{:} \odot \boldsymbol{Y}_{:}),\\
\boldsymbol{H} &= \boldsymbol{Q}_{:} \odot (\boldsymbol{V} \otimes \boldsymbol{I}_M),\\
\boldsymbol{K}_U &= \begin{bmatrix} \boldsymbol{K}_1^u & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{K}_2^u & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{K}_D^u \end{bmatrix},
\end{aligned} \tag{13}$$

and $\boldsymbol{Q} \in \mathbb{R}^{M\times N}$ is a binary indicator matrix with $q_{m,n} = 1$ if $(m, n) \in \Omega$ and $q_{m,n} = 0$ otherwise. We can use the same strategy for the factor matrix $\boldsymbol{V}$, by noting that:

$$\boldsymbol{Y}_{:}^\top = (\boldsymbol{U} \otimes \boldsymbol{I}_N)\,\boldsymbol{V}_{:} + \boldsymbol{E}_{:}^\top, \tag{14}$$

where $\boldsymbol{Y}_{:}^\top = \text{vec}(\boldsymbol{Y}^\top)$ and $\boldsymbol{E}_{:}^\top = \text{vec}(\boldsymbol{E}^\top)$. We obtain the following posterior for $\boldsymbol{V}$:

$$p\left(\boldsymbol{V}_{:} \mid \mathcal{P}_\Omega(\boldsymbol{Y}), \boldsymbol{U}, \tau, \boldsymbol{\vartheta}\right) \sim \mathcal{N}\left(\boldsymbol{V}_{:} \,\middle|\, \boldsymbol{\mu}_V^*, (\boldsymbol{\Lambda}_V^*)^{-1}\right), \tag{15}$$

where

$$\begin{aligned}
\boldsymbol{\Lambda}_V^* &= \tau\boldsymbol{H}^\top\boldsymbol{H} + \boldsymbol{K}_V^{-1},\\
\boldsymbol{\mu}_V^* &= \tau\left(\boldsymbol{\Lambda}_V^*\right)^{-1}\boldsymbol{H}^\top(\boldsymbol{Q}_{:}^\top \odot \boldsymbol{Y}_{:}^\top),\\
\boldsymbol{H} &= \boldsymbol{Q}_{:}^\top \odot (\boldsymbol{U} \otimes \boldsymbol{I}_N),
\end{aligned} \tag{16}$$

and $\boldsymbol{K}_V$ is a block diagonal matrix built from $\boldsymbol{K}_d^v$.

The posterior distributions of $\boldsymbol{U}$ and $\boldsymbol{V}$ can be factorized into the product of conditional distributions over the column vectors. For the $\boldsymbol{U}$ matrix, we have, for example:

$$\begin{aligned}
&p\left(\boldsymbol{U} \mid \mathcal{P}_\Omega(\boldsymbol{Y}), \boldsymbol{V}, \tau, \boldsymbol{\theta}\right)\\
&\qquad = \prod_{d=1}^{D} p\left(\boldsymbol{U}_{:d} \,\middle|\, \mathcal{P}_\Omega(\boldsymbol{Y}), \boldsymbol{U}_{:h,h\neq d}, \boldsymbol{V}, \tau, \boldsymbol{\theta}\right).
\end{aligned} \tag{17}$$

For each dimension d, the conditional distribution $p\left(\boldsymbol{U}_{:d} \mid -\right)$ is Gaussian:

$$p\left(\boldsymbol{U}_{:d} \,\middle|\, \mathcal{P}_\Omega(\boldsymbol{Y}), \boldsymbol{U}_{:h,h\neq d}, \boldsymbol{V}, \tau, \boldsymbol{\theta}\right) \sim \mathcal{N}\left(\boldsymbol{\mu}_d^*, (\boldsymbol{\Lambda}_d^*)^{-1}\right), \tag{18}$$

where

$$\begin{aligned}
\boldsymbol{\Lambda}_d^* &= \boldsymbol{\Upsilon}_d + \left(\boldsymbol{K}_d^u\right)^{-1},\\
[\boldsymbol{\Upsilon}_d]_{mm} &= \tau\sum_{n\in\Omega_m} v_{n,d}^2,\ m = 1, \cdots, M,\\
\boldsymbol{\mu}_d^* &= \left(\boldsymbol{\Lambda}_d^*\right)^{-1}\boldsymbol{\eta}_d,\\
\left[\boldsymbol{\eta}_d\right]_m &= \tau\sum_{n\in\Omega_m}\left(y_{m,n} - \sum_{h:h\neq d} u_{m,h}v_{n,h}\right)v_{n,d},\\
&\quad m = 1, \cdots, M,
\end{aligned}$$

and $\Omega_m = \{n : (m, n) \in \Omega\}$ denotes the set of indices n for which $y_{m,n}$ is observed. Note that $\boldsymbol{\Upsilon}_d$ is a $M \times M$ diagonal matrix and $\boldsymbol{\eta}_d$ is an $M \times 1$ column vector. The posterior distribution $p\left(\boldsymbol{V}_{:d} \mid -\right)$ can be obtained in the same way. These distributions can be sampled in parallel to speed up the sampling process. However, this strategy ignores the posterior dependencies among the columns of factors so the mixing can be slow.

2) Sampling Hyperparameters $\{\boldsymbol{\theta}, \boldsymbol{\vartheta}\}$: In the standard parameterization framework of BKMF (Figure 2), the hyperparameters $\{\boldsymbol{\theta}, \boldsymbol{\vartheta}\}$ are updated based on the model parameters $\{\boldsymbol{U}, \boldsymbol{V}\}$ and the hyper-priors. Taking the hyperparameters of $\boldsymbol{U}$ for example, the corresponding conditional posterior for θ in $\boldsymbol{\theta}_d$ is $p\left(\theta \mid \boldsymbol{U}_{:d}, \mu_\theta, \tau_\theta\right) \propto p\left(\boldsymbol{U}_{:d} \,\middle|\, \boldsymbol{0}, \boldsymbol{K}_d^u\right) p\left(\theta \,\middle|\, \mu_\theta, \tau_\theta^{-1}\right)$, which can be sampled using the Metropolis algorithm. However, conditioning on the latent factors to estimate the hyperparameters does not work in practice since the factors and the kernel hyperparameters are strongly coupled in such hierarchical models [16], [34]. To improve the sampling efficiency, we integrate out the latent factor variables ($\boldsymbol{U}$ or $\boldsymbol{V}$) from the model, and learn the hyperparameters from the marginal posteriors, such as $p\left(\theta \mid \boldsymbol{Y}\right) \propto p\left(\boldsymbol{Y} \mid \theta\right) p\left(\theta \,\middle|\, \mu_\theta, \tau_\theta^{-1}\right)$.

Specifically, to sample $\boldsymbol{\theta}_d$, we integrate over $\boldsymbol{U}_{:d}$ in Eq. (11) given the normal prior of $\boldsymbol{U}_{:d}$ ($p(\boldsymbol{U}_{:d}) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K}_d^u)$), obtaining the marginal likelihood:

$$\begin{aligned}
\log p\left(\mathcal{P}_\Omega(\boldsymbol{Y}) \middle| \boldsymbol{\theta}_d, \boldsymbol{U}_{:h,h\neq d}, \boldsymbol{V}, \tau\right) \propto &-\frac{1}{2}\boldsymbol{R}_{:}^\top\boldsymbol{\Sigma}_{\boldsymbol{y}|\theta}^{-1}\boldsymbol{R}_{:}\\
&-\frac{1}{2}\log\left|\boldsymbol{\Sigma}_{\boldsymbol{y}|\theta}\right|,
\end{aligned} \tag{19}$$

where $\boldsymbol{\Sigma}_{\boldsymbol{y}|\theta} = \boldsymbol{H}\boldsymbol{K}_d^u\boldsymbol{H}^\top + \tau\,\text{diag}(\boldsymbol{Q}_{:})$ with $\boldsymbol{H} = \boldsymbol{Q}_{:} \odot (\boldsymbol{V}_{:d} \otimes \boldsymbol{I}_M)$, $\boldsymbol{R} = \boldsymbol{Y} - \sum_{h:h\neq d}\boldsymbol{U}_{:h}\boldsymbol{V}_{:h}^\top$, and $\boldsymbol{R}_{:}$ denotes

Algorithm 1 Sampling for Each Hyperparameter θ in $\boldsymbol{\theta}_d$

Input: $\theta^{(k)}$, $\boldsymbol{U}^{(k)}$, $\boldsymbol{V}^{(k)}$, $\tau^{(k)}$, $\boldsymbol{Y}$, slice sampling scale ρ
Output: $\theta^{(k+1)}$

Generate an initial sampling range:
  $r \sim \text{Uniform}(0, \rho)$, $\theta_{min} = \theta^{(k)} - r$, $\theta_{max} = \theta_{min} + \rho$
Draw $\kappa \sim \text{Uniform}(0, 1)$
loop
  Draw proposal $\theta' \sim \text{Uniform}(\theta_{min}, \theta_{max})$
  Calculate $\left(\boldsymbol{K}_d^u\right)'$, $\boldsymbol{\Sigma}_{\boldsymbol{y}|\theta'}$, $\log p\left(\theta' \mid -\right)$ corresponding to θ'
  if $\exp\left\{\log p\left(\theta' \mid -\right) - \log p\left(\theta \mid -\right)\right\} > \kappa$ **then**
    return $\theta^{(k+1)} = \theta'$
    break
  else if $\theta' < \theta^{(k)}$ **then**
    Shrink sampling range minimum: $\theta_{min} = \theta'$
  else
    Shrink sampling range maximum: $\theta_{max} = \theta'$
  end if
end loop

Algorithm 2 Sampling Procedure for BKMF

Initialize model hyperparameters $\{\boldsymbol{\theta}^{(1)}, \boldsymbol{\vartheta}^{(1)}\}$ and parameters $\{\boldsymbol{U}^{(1)}, \boldsymbol{V}^{(1)}\}$
for $k = 1 : K$ **do**
  for $d = 1 : D$ **do**
    Sample each hyperparameter in $\boldsymbol{\theta}_d^{(k+1)}$ using Algorithm 1
  end for
  for $d = 1 : D$ **do**
    Sample each hyperparameter in $\boldsymbol{\vartheta}_d^{(k+1)}$ using Algorithm 1
  end for
  for $d = 1 : D$ **do**
    Update each factor column $\boldsymbol{U}_{:d}$ according to Eq. (18):
    $\boldsymbol{U}_{:d}^{(k+1)} \sim p\left(\boldsymbol{U}_{:d} \mid \boldsymbol{Y}, \boldsymbol{U}_{:h,h\neq d}^{(k)}, \boldsymbol{V}^{(k)}, \tau^{(k)}, \boldsymbol{\theta}_d^{(k+1)}\right)$
    Update each factor column $\boldsymbol{V}_{:d}$ similar to $\boldsymbol{U}_{:d}$:
    $\boldsymbol{V}_{:d}^{(k+1)} \sim p\left(\boldsymbol{V}_{:d} \mid \boldsymbol{Y}, \boldsymbol{V}_{:h,h\neq d}^{(k)}, \boldsymbol{U}^{(k+1)}, \tau^{(k)}, \boldsymbol{\vartheta}_d^{(k+1)}\right)$
  end for
  Sample model precision τ according to Eq. (21):
  $\tau^{(k+1)} \sim p\left(\tau \mid \boldsymbol{Y}, \boldsymbol{U}^{(k+1)}, \boldsymbol{V}^{(k+1)}, a_0, b_0\right)$
end for

$\text{vec}(\boldsymbol{R})$. We use the Woodbury matrix identity[1] to compute the inverse of $\boldsymbol{\Sigma}_{\boldsymbol{y}|\theta}$, and the first term in Eq. (19) becomes:

$$-\frac{1}{2}\boldsymbol{R}_{:}^{\top}\boldsymbol{\Sigma}_{\boldsymbol{y}|\theta}^{-1}\boldsymbol{R}_{:} \propto \frac{1}{2}\tau^2\boldsymbol{R}_{:}^{\top}\boldsymbol{H}\left((\boldsymbol{K}_d^u)^{-1} + \tau\boldsymbol{H}^{\top}\boldsymbol{H}\right)^{-1}\boldsymbol{H}^{\top}\boldsymbol{R}_{:}.$$

The second term of Eq. (19) related to the determinant of $\boldsymbol{\Sigma}_{\boldsymbol{y}|\theta}$ is computed based on the matrix determinant lemma[2] as:

$$-\frac{1}{2}\log\left|\boldsymbol{\Sigma}_{\boldsymbol{y}|\theta}\right| \propto -\frac{1}{2}\log\left|(\boldsymbol{K}_d^u)^{-1} + \tau\boldsymbol{H}^{\top}\boldsymbol{H}\right| - \frac{1}{2}\log\left|\boldsymbol{K}_d^u\right|.$$

The hyperparameters $\boldsymbol{\theta}_d$ are then updated with the marginal posteriors $\log p\left(\boldsymbol{\theta}_d \mid \mathcal{P}_{\Omega}(\boldsymbol{Y}), \boldsymbol{U}_{:h,h\neq d}, \boldsymbol{V}, \tau\right)$ as below:

$$\begin{aligned}\log p\left(\boldsymbol{\theta}_d \mid -\right) &\propto \log p\left(\mathcal{P}_{\Omega}(\boldsymbol{Y}) \mid \boldsymbol{\theta}_d, \boldsymbol{U}_{:h,h\neq d}, \boldsymbol{V}, \tau\right) \\ &\quad + \log p\left(\boldsymbol{\theta}_d\right) \\ &\propto \frac{1}{2}\tau^2\boldsymbol{R}_{:}^{\top}\boldsymbol{H}\left((\boldsymbol{K}_d^u)^{-1} + \tau\boldsymbol{H}^{\top}\boldsymbol{H}\right)^{-1}\boldsymbol{H}^{\top}\boldsymbol{R}_{:} \\ &\quad - \frac{1}{2}\log\left|(\boldsymbol{K}_d^u)^{-1} + \tau\boldsymbol{H}^{\top}\boldsymbol{H}\right| - \frac{1}{2}\log\left|\boldsymbol{K}_d^u\right| \\ &\quad + \log p\left(\boldsymbol{\theta}_d \mid \mu_{\theta}, \tau_{\theta}\right). \end{aligned} \tag{20}$$

Considering that the efficiency of classic Metropolis sampling is sensitive to the selection of the variances in the proposal distributions, we apply an adaptive search algorithm — slice sampling [18] — to sample hyperparameters according to Eq. (20). This strategy is shown to be more robust to the choice of the sampling parameters. The sampling process for each θ in $\boldsymbol{\theta}_d$ is given in Algorithm 1. The updating process for $\boldsymbol{\vartheta}_d$, i.e., the covariance hyperparameters of $\boldsymbol{V}_{:d}$, is similar.

3) Sampling the Precision Term τ: Assuming the model noise is isotropic, because of the conjugate prior setting, the posterior distribution of τ follows a Gamma distribution:

$$p\left(\tau \mid \boldsymbol{Y}, \boldsymbol{U}, \boldsymbol{V}, a_0, b_0\right) \sim \text{Gamma}\left(a^*, b^*\right), \tag{21}$$

where

$$\begin{aligned} a^* &= a_0 + \frac{1}{2}\sum_{(m,n)\in\Omega} q_{m,n}, \\ b^* &= b_0 + \frac{1}{2}\sum_{(m,n)\in\Omega}\left(y_{m,n} - \boldsymbol{U}_{m:}\boldsymbol{V}_{n:}^{\top}\right)^2. \end{aligned} \tag{22}$$

4) Implementation: The complete sampling procedure for BKMF is outlined in Algorithm 2. Note that when dealing with the kriging tasks, instead of the component-wise updating formulas shown in Algorithm 2, we sample the factor matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ as a whole using Eqs. (12) and (15) respectively to make full use of the dependencies in factors.

V. Experiments

To demonstrate the imputation and kriging performance of BKMF, we conduct numerical experiments using real-world spatiotemporal traffic datasets under various missing scenarios. We compare results with several state-of-the-art spatiotemporal analysis models. One of the key strengths of BKMF is its ability to automatically learn kernel hyperparameters from the data. This allows us to fully explore the performance of different kernel prior settings on spatiotemporal pattern representation and discovery. Therefore, several alternative kernel priors will also be tested and evaluated in the experiments.

A. Data Description and Kernel Introduction

We conduct experiments on two publicly available traffic datasets, both containing spatial structure information: **(1) SeData**,[3] a traffic speed dataset collected from 323 loop detectors in Seattle highway over one year at 5-min interval [35]. Here, we use one month of data and decrease data size by selecting every 12th observation (i.e., one point per hour). The selected raw data contains no missing values. **(2) METR-LA**[4] is also a traffic speed dataset, which includes

[1]For matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$, $\left(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{C}\boldsymbol{B}^{\top}\right)^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{B}\left(\boldsymbol{C}^{-1} + \boldsymbol{B}^{\top}\boldsymbol{A}^{-1}\boldsymbol{B}\right)^{-1}\boldsymbol{B}^{\top}\boldsymbol{A}^{-1}$.

[2]$\left|\boldsymbol{A} + \boldsymbol{B}\boldsymbol{C}\boldsymbol{B}^{\top}\right| = \left|\boldsymbol{C}^{-1} + \boldsymbol{B}^{\top}\boldsymbol{A}^{-1}\boldsymbol{B}\right| \det(\boldsymbol{C}) \det(\boldsymbol{A})$.

[3]https://github.com/zhiyongc/Seattle-Loop-Data

[4]https://github.com/liyaguang/DCRNN

TABLE II
PRIOR KERNEL FUNCTIONS FOR TEMPORAL DIMENSION

Kernel function	Formulation
Exponential	$k_{Exp}(\Delta; l_t, \sigma_t) = \sigma_t^2 \exp\left\{-\frac{\Delta}{l_t}\right\}$
Matern 3/2	$k_{\nu=3/2}(\Delta; l_t, \sigma_t) = \sigma_t^2 \left(1+\frac{\sqrt{3}\Delta}{l_t}\right) \exp\left\{-\frac{\sqrt{3}\Delta}{l_t}\right\}$
Matern 5/2	$k_{\nu=5/2}(\Delta; l_t, \sigma_t) = \sigma_t^2 \left(1+\frac{\sqrt{5}\Delta}{l_t}+\frac{5\Delta^2}{3l_t^2}\right) \exp\left\{-\frac{\sqrt{5}\Delta}{l_t}\right\}$
SE	$k_{SE}(\Delta; l_t, \sigma_t) = \sigma_t^2 \exp\left\{-\frac{\Delta^2}{2l_t^2}\right\}$

speed observations collected by highway loop detectors in Los Angeles [10]. We select data over ten days (May 1, 2012 to May 10, 2012) from 207 sensors and select every 4th point (i.e., one point per 20-min interval), among which 1.32% of the points are missing in the raw dataset.

For both datasets, we test two types of graph kernels, regularized Laplacian and diffusion kernel [32], [36], as the prior kernel function for the spatial factors in our model. To build these graph kernels, we first define a symmetric weighted matrix $\boldsymbol{W}$ using the exponential function based on the road network distances:

$$w_{i,j} = \exp\left\{-\frac{\text{dist}\,(i,\,j)^2}{l_s^2}\right\}, \tag{23}$$

where $\text{dist}(i, j)$ is the shortest road network distance between the i-th and j-th sensors, l_s is the spatial lengthscale hyperparameter. Note that the diagonal entries of $\boldsymbol{W}$ are zeros. The Laplacian matrix is then calculated as $\boldsymbol{Lap} = \text{diag}\,(\boldsymbol{W}\boldsymbol{1}_M) - \boldsymbol{W}$. Based on the $\boldsymbol{Lap}$ matrix, we define the regularized Laplacian (k_{RL}) and the diffusion kernel (k_{Diff}) as below:

$$\begin{aligned} k_{RL}(\boldsymbol{Lap}; \beta) &= (\boldsymbol{I}_M + \beta \boldsymbol{Lap})^{-1}, \\ k_{Diff}(\boldsymbol{Lap}; \beta) &= \exp\left(-\beta \boldsymbol{Lap}\right), \end{aligned} \tag{24}$$

where β is the kernel hyperparameter.

For the temporal factors, there are several choices for the GP prior setting [12]. We list the kernel functions considered in this work in Table II, where $\Delta = t_i - t_j$ denotes the distance between time indices i and j, l_t is the lengthscale hyperparameter for temporal dimension, and σ_t is the variance hyperparameter reflecting the magnitude of factors.

B. Experiment Settings

We perform imputation and kriging experiments on SeData and METR-LA. For the imputation task, we test 50%, 70%, 90% and 95% random missing (RM) for both datasets. For the kriging task, we set 20%, 40% sensors as unknown locations with no observations. Then as illustrated in Figure 1, for the time series collected from existing sensors, we further introduce three missing patterns: (a) randomly putting 50% of the sampled data as missing (kriging with random missing, K-RM for short); (b) first selecting 50% RM for the sampled signals then choosing 40% random block missing on the remained data (kriging with random block missing, denoted as K-RB), where the block sizes for SeData and METR-LA are 12 and 18 time points, respectively; (c) first selecting 50% RM then choosing 40% nonrandom block missing (kriging with nonrandom block missing, written as K-NB), with block size being 6 and 8 for SeData and METR-LA. K-RM, K-RB, K-NB correspond to the scenarios shown in Figure 1 (a), (b), (c), respectively. The locations of unknown sensors are randomly selected among the sensors with at least one adjacent node to ensure that the spatial structure information can contribute to the kriging process. Note that when most of the sensors are missing, e.g., in citywide estimation, additional spatial information is needed for inferring data at locations without nearby adjacent nodes, such as the work in [29].

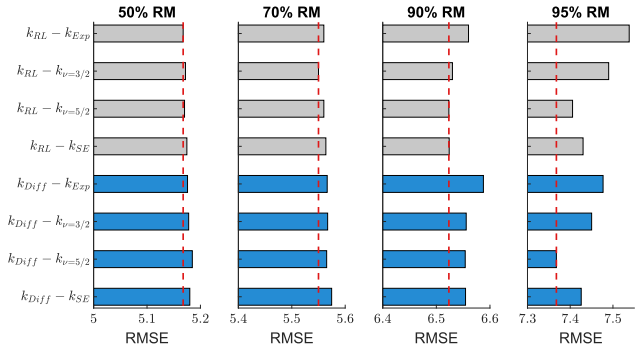
To explore the kernel representation properties, we test different kernel function settings. As explained in the last section, we compare eight settings of spatiotemporal kernels ($k_{spatial} - k_{temporal}$) following Eq. (24) and Table II: $\left\{k_{RL} - k_{Exp}, k_{RL} - k_{\nu=3/2}, k_{RL} - k_{\nu=5/2}, k_{RL} - k_{SE}\right\}$, and $\left\{k_{Diff} - k_{Exp}, k_{Diff} - k_{\nu=3/2}, k_{Diff} - k_{\nu=5/2}, k_{Diff} - k_{SE}\right\}$. Specifically, for spatial factors, we apply the same kernel with hyperparameters l_s and β; for temporal factors, we use D kernels with the same scale hyperparameter l_t and respective variance hyperparameters $\{\sigma_d : d = 1, \cdots, D\}$. The factors $\{\boldsymbol{U}, \boldsymbol{V}\}$ are initialized randomly by sampling from a standard normal distribution, and the unknown hyperparameters are initialized to 1. For the imputation task, the rank D is fixed to 15, 10, 5, 5, for 50%, 70%, 90%, 95% RM, respectively. For the kriging task, we assume $D = 10$ and $D = 5$ for 20% and 40% sensor missing, respectively. For MCMC sampling, we use 2000 iterations to make sure that hyperparameters and parameters converge, where 500 iterations are taken as burn-in.

C. Competing Models

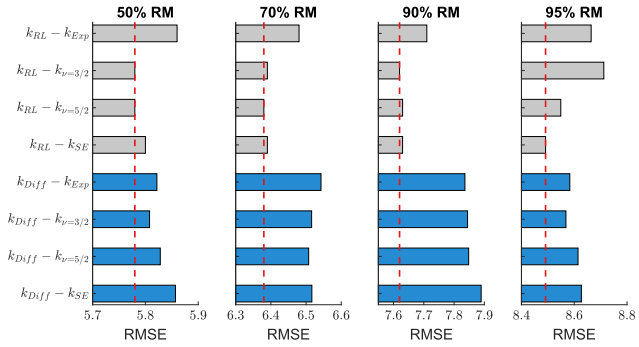
To demonstrate the advantage of the proposed framework, we compare BKMF with several widely used spatiotemporal analysis models named in Table I:

- *Graph Regularized Methods:* **GSMF** (Graph Structured MF model) [7], where the graph Laplacian regularization constrains both spatial and temporal factors. **GLTL** (Greddy Low-rank Tensor Learning) [3], a tensor factorization approach for spatial cokriging, where a graph Laplacian is used to capture spatial correlations.
- *AR Regularized Methods:* We consider **BPTF** (Bayesian Probabilistic Tensor Factorization), a Bayesian tensor factorization model with one-order dynamic constraint for temporal collaborative filtering [9], and **TRMF** (Temporal Regularized Matrix Factorization), a MF model with temporal AR regularization for high-dimensional time series imputation and prediction [5]. Note that BPTF and TRMF can perform well in RM imputation tasks, but they cannot deal with spatial interpolation, i.e., kriging.
- *GP Regularized Methods:* **GPFA** (Variational GP Factor Analysis) [14], a GP constrained MF model which uses variational inference to approximate the posteriors of latent factors. KPMF is also a representative approach, but we do not include it in the baselines since the hyperparameters are difficult to decide.

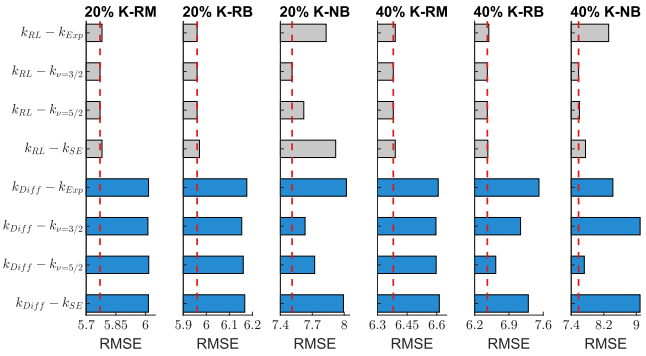
For TRMF, we assume the lag indices are $\{1, \cdots, 6\}$ for both datasets and choose temporal regularization parameters



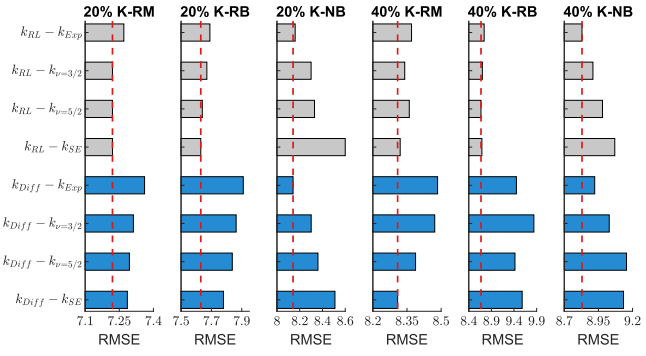
(a) Kernel comparison for *imputation* of SeData.



(b) Kernel comparison for *imputation* of METR-LA.



(c) Kernel comparison for *kriging* of SeData.



(d) Kernel comparison for *kriging* of METR-LA.

Fig. 3. Performance comparison of BKMF with different kernel settings for the imputation and kriging tasks (in RMSE), where the vertical red dash line denotes the best RMSE value.

through cross-validation. For graph regularized GSMF and GLTL, we use the same Laplacian matrices as in BKMF and select the regularization parameters through cross-validation. For VGFA, we model both spatial and temporal factors with the SE kernel following [14]. The model rank of all

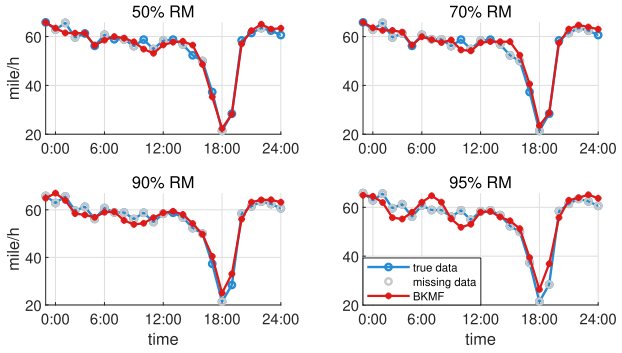


Fig. 4. RM imputation results of Sensor #212 on Day 5 for SeData, in which the results of BKMF giving the best performance are shown.

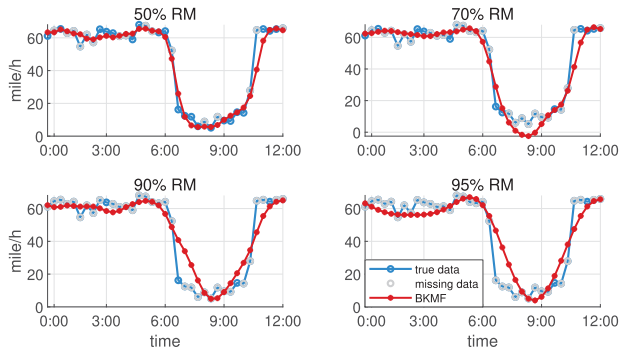


Fig. 5. RM imputation results of Sensor #89 on Day 2 for METR-LA.

these competing approaches is set as being the same as in BKMF.

D. Evaluation Metrics

To evaluate the performance of the methods, we compute two metrics: mean absolute error (MAE) and root mean square error (RMSE), defined as follows:

$$\text{MAE} = \frac{1}{n}\sum_{i\notin\Omega}\left|y_i - \hat{y}_i\right|,$$
$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i\notin\Omega}\left(y_i - \hat{y}_i\right)^2}, \tag{25}$$

where n is the number of missing values, y_i and $\hat{y}_i$ are the true value and estimation of the i-th missing element, respectively.

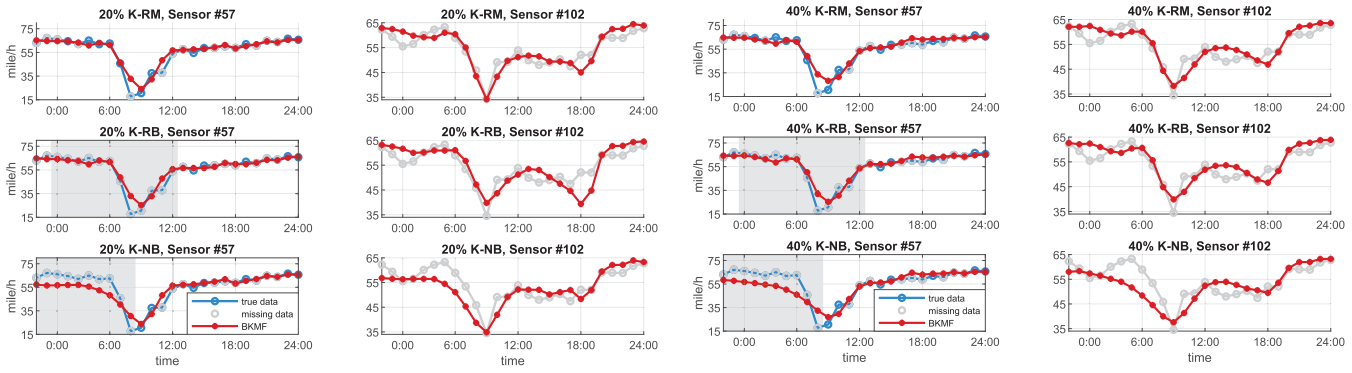
E. Results and Analysis

1) Random Missing Imputation: We first compare the imputation RMSE of BKMF with various kernel priors in Figure 3 (a) and (b). We can see that different kernel combinations achieve similar accuracy in 50% and 70% RM scenarios. This result indicates the flexibility of GP regularization since any reasonable kernel priors are able to capture the spatiotemporal correlations even when more than half of the data is missing. The choice of kernel becomes important when the data is extremely sparse, e.g., in the case of 95% RM, where smoother temporal kernels tend to provide better

TABLE III
RM IMPUTATION AND SPATIOTEMPORAL KRIGING PERFORMANCE (MAE/RMSE)

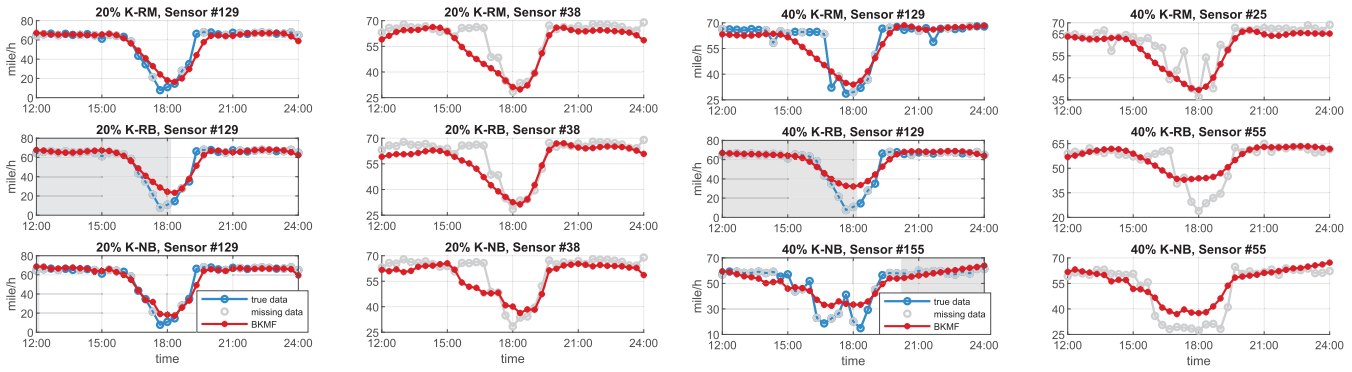
Data	Scenarios	GSMF	GLTL	BPTF	TRMF	VGFA	BKMF
(S)	50% RM	3.82/5.82	5.50/8.71	3.23/5.23	3.58/5.71	3.62/5.53	**3.18/5.17**
	70% RM	4.23/6.44	6.18/9.76	3.51/5.69	3.83/6.06	3.82/5.88	**3.41/5.55**
	90% RM	4.96/7.54	6.88/10.85	4.21/6.88	4.79/7.19	4.32/6.69	**3.99/6.52**
	95% RM	5.28/7.99	7.09/11.09	5.01/8.45	5.97/8.45	5.36/7.90	**4.51/7.37**
	20% K-RM	5.01/8.13	5.42/8.61	-	-	4.11/6.19	**3.63/5.77**
	20% K-RB	4.76/7.74	6.15/9.69	-	-	4.33/6.61	**3.76/5.96**
	20% K-NB	5.27/8.41	6.20/9.94	-	-	6.18/9.83	**4.65/7.51**
	40% K-RM	5.92/9.54	5.39/8.60	-	-	4.28/6.59	**4.09/6.38**
	40% K-RB	5.67/9.18	6.13/9.71	-	-	4.51/6.83	**4.12/6.46**
	40% K-NB	6.20/9.97	6.10/9.82	-	-	6.19/9.50	**4.79/7.58**
(M)	50% RM	5.02/7.51	6.86/10.05	3.61/5.88	3.63/5.93	4.08/6.59	**3.54/5.78**
	70% RM	5.26/7.70	7.63/11.03	4.06/6.54	4.09/6.49	4.69/7.50	**3.96/6.38**
	90% RM	6.05/8.83	8.36/12.10	4.91/7.91	5.39/8.00	5.86/9.06	**4.77/7.62**
	95% RM	6.28/9.16	8.48/12.38	5.89/9.79	7.00/9.41	6.60/9.90	**5.08/8.49**
	20% K-RM	6.24/9.35	7.42/10.72	-	-	5.69/9.18	**4.53/7.22**
	20% K-RB	6.32/9.37	8.06/11.44	-	-	5.77/9.23	**4.71/7.63**
	20% K-NB	7.23/10.53	8.06/11.23	-	-	6.01/9.25	**5.09/8.14**
	40% K-RM	7.16/10.60	7.47/10.69	-	-	6.45/10.02	**5.45/8.31**
	40% K-RB	7.19/10.65	7.96/11.49	-	-	6.21/9.88	**5.56/8.67**
	40% K-NB	7.39/10.59	8.04/11.27	-	-	7.56/10.87	**5.65/8.83**

Best results are highlighted in bold fonts.



(a) 20% kriging for SeData.

(b) 40% kriging for SeData.



(c) 20% kriging for METR-LA.

(d) 40% kriging for METR-LA.

Fig. 6. Kriging results for SeData and METR-LA, where the gray background denotes the consecutive temporal block missing. **(a)** and **(b)** show the results of SeData, including the estimated signals of one sampled sensor (Sensor #57) and one unknown sensor (Sensor #102). **(c)** and **(d)** are the results for METR-LA, where the selected unknown sensor for illustration is different between tasks because the unsampled sensors are different in 20% and 40% kriging for this data. In each case, the result of BKMF with the best RMSE is given.

performances. Specifically, BKMF with $k_{Diff} - k_{\nu=5/2}$ and $k_{RL} - k_{SE}$ performed best in 95% RM for SeData and METR-LA, respectively.

The comparison of imputation performances with other baseline methods is given in Table III, in which we show the results of BKMF having the best RMSE. It is clear that our

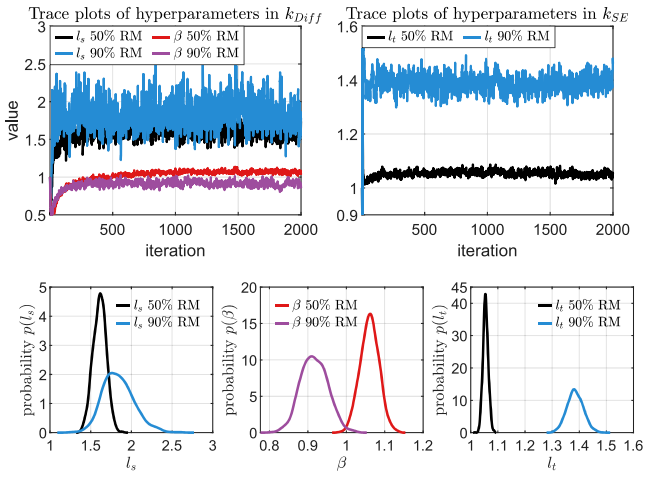
proposed BKMF consistently achieves the best performance, obtaining the lowest error in all scenarios. The superiority of BKMF becomes increasingly salient when the missing data increases, which implies that the Bayesian framework enables BKMF to be less sensitive to the increasing missing rates. Among the comparing methods, GSMF and GLTL have poor results indicating that the graph regularization may not effectively learn the temporal dependencies, and the cross-validation cannot ensure a local fine-grained solution for the model hyperparameters. BPTF and TRMF both perform well when the missing rate is relatively low and the rank/parameters are appropriately selected, but fail when the number of observations is small or the parameters are not fully tuned. VGFA also obtains large errors in several missing cases, which indicates that the kernel hyperparameters may not be well approximated.

We show the imputed time series of a randomly selected sensor in Figures 4 and 5 for SeData and METR-LA, respectively. We see that with the GP regularized low-rank framework, BKMF learns both global trends and local temporal dynamics in the data. Even under the 95% RM scenario where sensors capture very few observations, BKMF can still estimate missing values as the Bayesian modeling is free from parameter tuning and both spatial and temporal correlations play positive effects.

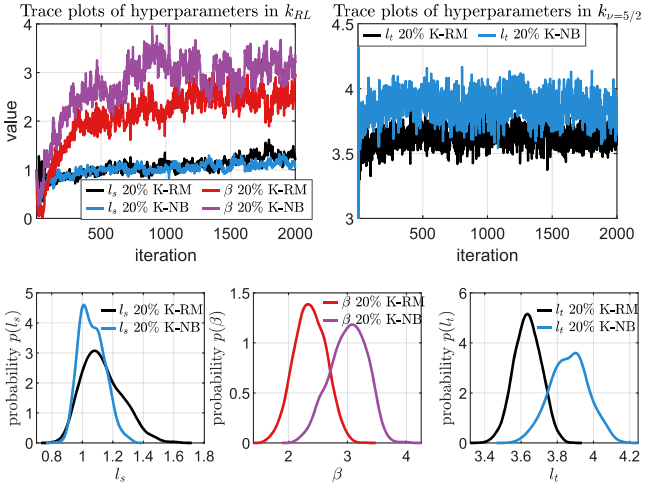
2) Spatiotemporal Kriging: The kriging RMSE of different kernel settings are also compared in Figure 3 ((c) and (d)). The comparison shows that the regularized Laplacian kernel performs better than the diffusion kernel in most of the kriging tasks. This implies that the inverse-form regularized Laplacian, which usually generates functions varying more rapidly compared to the exponential diffusion kernel, can capture more exact variations/dynamics between locations, which is crucial for spatial interpolation (i.e., kriging). Furthermore, the model is sensitive to the choice of kernel when there exists complex block missing, particularly in 40% K-NB. Benefiting from the Bayesian sampling scheme, BKMF allows us to fairly compare different kernel combinations without prior knowledge, and to further investigate kernel performances and choose the optimal kernel priors in specific scenarios. As for the comparison with baselines, we still summarize the kriging MAE/RMSE results in Table III. It is evident that BKMF outperforms other competing approaches with clearly lower errors. An interesting finding is that GLTL produces better results under 20% or 40% K-RM than pure 50% RM for SeData, which highlights the importance of spatial correlation in kriging tasks.

We illustrate some of the kriging results in Figure 6 for the two datasets. The estimations show that BKMF is able to effectively reconstruct time series for the unknown sensors, even when the observed sensors have a complicated missing pattern. Specifically, BKMF can estimate the sudden speed drop of morning or evening peak in 20% and 40% K-RM. When some consecutive temporal points are missing over the neighborhood or all sensors (K-RB/K-NB), BKMF still recovers the temporal trend of the unknown sensor, despite lacking the spatial information from adjacent locations.

3) Kernel Hyperparameters Illustration: We visualize some examples of the kernel hyperparameters (i.e., $\{l_s, \beta, l_t\}$) learned for imputation and kriging in Figure 7 (a) and (b), respectively, including the hyperparameter trace plots and the corresponding probability distributions after burn-in. Specifically, Panel (a) gives the results of l_s, β in k_{Diff} and l_t in k_{SE} for 50% RM and 90% RM imputation on SeData, and Panel (b) plots the results of l_s, β in k_{RL} and l_t in $k_{\nu=5/2}$ for 2 kriging cases (20% K-RM and 20% K-NB) on METR-LA. The plots show that the Markov chains for kernel hyperparameters mix well and that we can get interpretable distributions of hyperparameters. For example, the temporal lengthscale l_t becomes larger in 90% RM compared to 50% RM, meaning that the kernel functions change more smoothly in time when the data is more sparse.



(a) Kernel hyperparameters of $k_{Diff} - k_{SE}$ for SeData imputation.



(b) Kernel hyperparameters of $k_{RL} - k_{\nu=5/2}$ for METR-LA kriging.

Fig. 7. Trace plots and probability distributions of hyperparameters in corresponding spatial and temporal kernels for imputation on SeData and kriging on METR-LA. The kernel settings are $k_{Diff} - k_{SE}$ and $k_{RL} - k_{\nu=5/2}$ in (a) and (b), respectively.

VI. Conclusion and Discussion

A. Conclusion

In this paper, we propose a new Bayesian kernelized probabilistic MF (BKMF) model for spatiotemporal traffic data

imputation and kriging. We introduce GP priors over the columns of factors in the MF framework and develop an MCMC sampling algorithm for model inference. The model can characterize both global and local spatiotemporal structures of the data effectively. Particularly, we explore the impact of selecting different kernel functions as the prior on both the imputation and kriging tasks. Compared to other low-rank spatiotemporal models, BKMF achieves the best estimation performance for real traffic datasets and it is more robust high sparsity.

B. Connections to Existing Low-Rank Spatiotemporal Models

BKMF is a generalization of many existing low-rank models for spatiotemporal data analysis. Specifically, in terms of graph structure, GSMF [7] corresponds to applying a regularized Laplacian kernel prior to all the decomposed factors. As for temporal dimension, BPTF [9] is equivalent to placing the Ornstein-Uhlenbeck (OU) process as the covariance function of temporal factors, and TRMF [5] equals to using Matern class kernel functions as the temporal prior [12]. GLTL [3] and several other spatiotemporal models such as in [8], [24], can be seen as combining Laplacian kernel and Matern kernels for spatial and temporal latent components, respectively.

C. Advantages and Limitations

Three notable advantages of this work are:

1) We propose a Bayesian sampling framework for the spatiotemporal GP regularized MF model which can efficiently learn both latent factors and kernel hyperparameters. The Bayesian framework and kernel properties allow us to automatically account for the complex spatiotemporal dependencies without much prior knowledge.
2) In our low-rank GP model, there is no need to consider a complicated composite covariance function. Using a simple type kernel form such as Matern 3/2 as the GP prior of factors is enough to learn the data correlation, since the kernel constraint captures local smoothness/dynamics and the global consistency such as the daily/weekly trend will be learned by the low-rank assumption.
3) The results of BKMF are interpretable and can be utilized to analyze spatiotemporal traffic patterns in practice. The learned kernel hyperparameters can extrapolate future correlations in both the space and time domains.

The main limitation of BKMF is still the computational cost. The inference process for kernel hyperparameters and latent factors takes time and memory when the data size is large. An approach to reduce the computation load is to use sparse GP approximations by introducing inducing points [37] or compact support kernels [38]. One can also apply low-rank approximation for the factor covariance matrices [39] to further decrease the computational time.

REFERENCES

[1] S. Shekhar *et al.*, "Spatiotemporal data mining: A computational perspective," *ISPRS Int. J. Geo-Inf.*, vol. 4, no. 4, pp. 2306–2338, Oct. 2015.
[2] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 3634–3640.
[3] M. T. Bahadori, Q. R. Yu, and Y. Liu, "Fast multivariate spatio-temporal analysis via low rank tensor learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 3491–3499.
[4] Y. Wu, D. Zhuang, A. Labbe, and L. Sun, "Inductive graph neural networks for spatiotemporal Kriging," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 5, pp. 4478–4485.
[5] H.-F. Yu, N. Rao, and I. S. Dhillon, "Temporal regularized matrix factorization for high-dimensional time series prediction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 847–855.
[6] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. 20th Int. Conf. Neural Inf. Process. Syst.*, 2008, pp. 1257–1264.
[7] N. Rao, H.-F. Yu, P. K. Ravikumar, and I. S. Dhillon, "Collaborative filtering with graph information: Consistency and scalable methods," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 2107–2115.
[8] Y. Wang, Y. Zhang, X.-L. Piao, H. Liu, and K. Zhang, "Traffic data reconstruction via adaptive spatial-temporal correlations," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 4, pp. 1531–1543, Aug. 2018.
[9] L. Xiong, X. Chen, T.-K. Huang, J. Schneider, and J. G. Carbonell, "Temporal collaborative filtering with Bayesian probabilistic tensor factorization," in *Proc. SIAM Int. Conf. Data Mining*, Apr. 2010, pp. 211–222.
[10] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.
[11] L. Li, X. Su, Y. Zhang, Y. Lin, and Z. Li, "Trend modeling for traffic time series analysis: An integrated study," *Intell. Transp. Syst.*, vol. 16, no. 6, pp. 3430–3439, Dec. 2015.
[12] C. K. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*. Oxfordshire, U.K.: Taylor & Francis, 2006.
[13] S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson, and S. Aigrain, "Gaussian processes for time-series modelling," *Philos. Trans. Roy. Soc. London A, Math. Phys. Sci.*, vol. 371, no. 1984, 2013, Art. no. 20110550.
[14] J. Luttinen and A. Ilin, "Variational Gaussian-process factor analysis for modeling spatio-temporal data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1177–1185.
[15] M. Filippone, M. Zhong, and M. Girolami, "A comparative evaluation of stochastic-based inference methods for Gaussian process models," *Mach. Learn.*, vol. 93, no. 1, pp. 93–114, Oct. 2013.
[16] O. Papaspiliopoulos, G. O. Roberts, and M. Sköld, "A general framework for the parametrization of hierarchical models," *Stat. Sci.*, vol. 22, no. 1, pp. 59–73, Feb. 2007.
[17] M. Filippone and M. Girolami, "Pseudo-marginal Bayesian inference for Gaussian processes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 11, pp. 2214–2226, Nov. 2014.
[18] R. M. Neal, "Slice sampling," *Ann. Statist.*, vol. 31, no. 3, pp. 705–741, Jun. 2003.
[19] D. G. Krige, "A statistical approach to some basic mine valuation problems on the Witwatersrand," *J. Chem.*, vol. 52, no. 6, pp. 119–139, Dec. 1951.
[20] E. H. Isaaks and M. R. Srivastava, *Applied Geostatistics*. Oxford, U.K.: Oxford Univ. Press, 1989.
[21] B. Shamo, E. Asa, and J. Membah, "Linear spatial interpolation and analysis of annual average daily traffic data," *J. Comput. Civil Eng.*, vol. 29, no. 1, Jan. 2015, Art. no. 04014022.
[22] Y. Yang, J. Wu, and G. Christakos, "Prediction of soil heavy metal distribution using spatiotemporal Kriging with trend model," *Ecol. Indicators*, vol. 56, pp. 125–133, Sep. 2015.
[23] C. J. Ruybal, T. S. Hogue, and J. E. McCray, "Evaluation of groundwater levels in the Arapahoe aquifer using spatiotemporal regression Kriging," *Water Resour. Res.*, vol. 55, no. 4, pp. 2820–2837, Apr. 2019.
[24] R. Yu and Y. Liu, "Learning from multiway data: Simple and efficient tensor regression," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 373–381.
[25] X. Chen and L. Sun, "Bayesian temporal factorization for multidimensional time series prediction," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Mar. 17, 2021, doi: 10.1109/TPAMI.2021.3066551.
[26] X. Chen, M. Lei, N. Saunier, and L. Sun, "Low-rank autoregressive tensor completion for spatiotemporal traffic data imputation," *IEEE Trans. Intell. Transp. Syst.*, early access, Sep. 27, 2021, doi: 10.1109/TITS.2021.3113608.
[27] T. Zhou, H. Shan, A. Banerjee, and G. Sapiro, "Kernelized probabilistic matrix factorization: Exploiting graphs and side information," in *Proc. SIAM Int. Conf. Data Mining*, Apr. 2012, pp. 403–414.

[28] X. Yi, Z. Duan, T. Li, T. Li, J. Zhang, and Y. Zheng, "CityTraffic: Modeling citywide traffic via neural memorization and generalization approach," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Nov. 2019, pp. 2665–2671.
[29] Z. Zhang, M. Li, X. Lin, and Y. Wang, "Network-wide traffic flow estimation with insufficient volume detection and crowdsourcing data," *Transp. Res. C, Emerg. Technol.*, vol. 121, Dec. 2020, Art. no. 102870.
[30] Y. Yu, X. Tang, H. Yao, X. Yi, and Z. Li, "Citywide traffic volume inference with surveillance camera records," *IEEE Trans. Big Data*, vol. 7, no. 6, pp. 900–912, Dec. 2021.
[31] R. Salakhutdinov and A. Mnih, "Bayesian probabilistic matrix factorization using Markov chain Monte Carlo," in *Proc. 25th Int. Conf. Mach. Learn. (ICML)*, 2008, pp. 880–887.
[32] A. J. Smola and R. Kondor, "Kernels and regularization on graphs," in *Learning Theory and Kernel Machines*. Berlin, Germany: Springer, 2003, pp. 144–158.
[33] I. Murray and R. P. Adams, "Slice sampling covariance hyperparameters of latent Gaussian models," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 23, 2010, pp. 1732–1740.
[34] Y. Yu and X.-L. Meng, "To center or not to center: That is not the question—An ancillarity–sufficiency interweaving strategy (ASIS) for boosting MCMC efficiency," *J. Comput. Graph. Statist.*, vol. 20, no. 3, pp. 531–570, Jan. 2011.
[35] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction," 2018, *arXiv:1801.02143*.
[36] F. Fouss, L. Yen, A. Pirotte, and M. Saerens, "An experimental investigation of graph kernels on a collaborative recommendation task," in *Proc. 6th Int. Conf. Data Mining (ICDM)*, Dec. 2006, pp. 863–868.
[37] J. Quiñonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *J. Mach. Learn. Res.*, vol. 6, pp. 1939–1959, Dec. 2005.
[38] J. Luttinen and A. Ilin, "Efficient Gaussian process inference for short-scale spatio-temporal modeling," in *Proc. 15th Int. Conf. Artif. Intell. Statist.*, 2012, pp. 741–750.
[39] C. Williams, E. V. Bonilla, and K. M. Chai, "Multi-task Gaussian process prediction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 153–160.

Mengying Lei received the B.S. degree in automation from Huazhong Agricultural University in 2016 and the M.S. degree from the School of Automation Science and Electrical Engineering, Beihang University, Beijing, China, in 2019. She is currently pursuing the Ph.D. degree with the Department of Civil Engineering, McGill University, Montreal, QC, Canada. Her research currently focuses on spatiotemporal data modeling and intelligent transportation systems.

Aurelie Labbe received the Ph.D. degree in statistics from the University of Waterloo, Canada, in 2005. She is currently a Full Professor at the Department of Decision Science, HEC Montréal. She holds the FRQ-IVADO Research Chair in data science. Her research interests include statistical learning, Bayesian and spatiotemporal modeling, data dimension reduction strategies, and complex correlation structures.

Yuankai Wu received the Ph.D. degree from the School of Mechanical Engineering, Beijing Institute of Technology, Beijing, China, in 2019. He was a Visiting Ph.D. Student with the Department of Civil and Environmental Engineering, University of Wisconsin–Madison, from November 2016 to November 2017. He is currently a Post-Doctoral Researcher with the Department of Civil Engineering, McGill University, supported by the Institute For Data Valorization (IVADO). His research interests include intelligent transportation systems, intelligent energy management, and machine learning.

Lijun Sun (Member, IEEE) received the B.S. degree in civil engineering from Tsinghua University, Beijing, China, in 2011, and the Ph.D. degree in civil engineering (transportation) from the National the University of Singapore in 2015. He is currently an Assistant Professor with the Department of Civil Engineering, McGill University, Montreal, QC, Canada. His research centers on intelligent transportation systems, machine learning, spatiotemporal modeling, travel behavior, and agent-based simulation.