Communications in Statistics - Simulation and Computation

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/lssp20

Structural learning of functional directed graphical models with incomplete signals

Dengyu Li & Kaibo Wang

To cite this article: Dengyu Li & Kaibo Wang (02 Aug 2023): Structural learning of functional directed graphical models with incomplete signals, Communications in Statistics - Simulation and Computation, DOI: 10.1080/03610918.2023.2234675

To link to this article: https://doi.org/10.1080/03610918.2023.2234675

COMMUNICATIONS IN STATISTICS - SIMULATION AND COMPUTATION®
https://doi.org/10.1080/03610918.2023.2234675

Structural learning of functional directed graphical models with incomplete signals

Dengyu Li[a] and Kaibo Wang[a,b]

[a]Department of Industrial Engineering, Tsinghua University, Beijing, China; [b]Vanke School of Public Health, Tsinghua University, Beijing, China

ABSTRACT

Functional directed graph model (DGM) learning has been widely used to analyze complex systems. Most existing works assume that the functional signals are complete, which in reality is not true. To address this problem, in this study, a framework for DGM learning with incomplete signals is proposed. Specifically, a penalty term that integrates information from the graph structure is added to the Maximum Margin Matrix Factorization (MMMF) objective function. The proposed method can be used with a known structure to estimate the functional relationship between nodes or with an unknown structure to estimate the relationship together with the graph structure. Numerical experiments and a real-world case study of monocrystalline silicone manufacturing are performed to verify the effectiveness of the proposed method when the signal matrices are incomplete.

ARTICLE HISTORY
Received 27 September 2022
Accepted 4 July 2023

KEYWORDS
Functional directed graph models; function-to-function regression; graph learning; missing data; signal completion

1. Introduction

In various fields, such as environmental concerns, cognitive maintenance, supply chain and manufacturing, systems have become increasingly complex due to the emergence of numerous components and the relationships among them (Langseth and Portinale 2007; Cao, Sandstede, and Luo 2019; Tan, Zhang, and Cai 2019; Estrada, Paynabar, and Pacella 2021; Guo and Zhang 2021; Han et al. 2021; Xia et al. 2022). Many applications have shown that the DGM is a useful tool that can provide a probabilistic representation of these complex systems, in which nodes represent components or entities and directed arcs representing functional relationships between nodes (Tan, Zhang, and Cai 2019; Estrada, Paynabar, and Pacella 2021).

Although most of the DGMs assume the observations of nodes to be scalars, the developing sensor technology has improved the accuracy and amount of data collected from components of systems, which makes functional observations of nodes available. Functional observations are collected in signal form with high fidelity and frequency. However, in real-world applications, for various reasons such as reading/writing errors, sensor faults, or interruptions during long processing cycles, the actual collected signals are sometimes incomplete (Fang et al. 2021; Yan et al. 2022). In this work, we focus on scenarios in which functional signals are partially observed as shown in Figure 1. Note that for convenience, the shown pattern involves random absence, but several other patterns of missed data, such as non-uniform missing (continuous missing) and imbalanced missing (different proportions for each signal) data, are considered in this study, and a detailed description can be found in Sec. 3. The existence of incomplete

CONTACT Kaibo Wang kbwang@tsinghua.edu.cn Vanke School of Public Health, Tsinghua University, Beijing 100084, China.

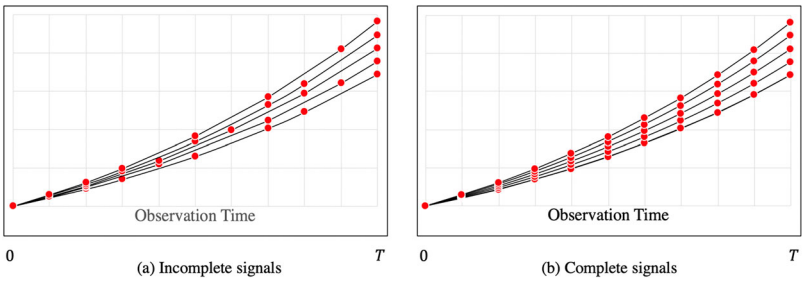

Figure 1. A simplified illustration of the differences between incomplete signals and complete signals (red points represent the observed data, and lines represent the potential curves).

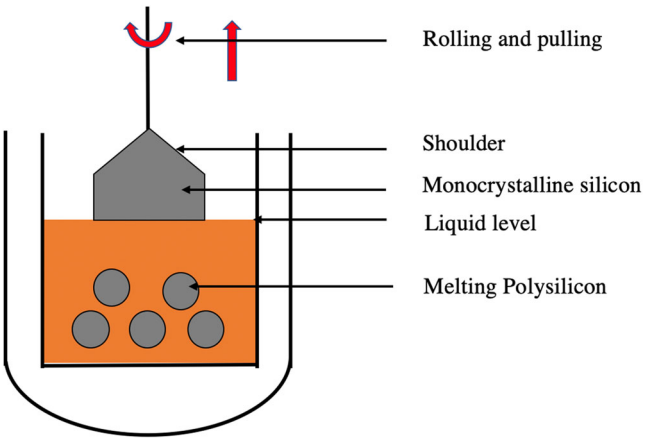

Figure 2. A simplified view of the CZ method as used in the monocrystalline silicon manufacturing process.

signals is a challenging problem when learning DGMs because simply deleting observations consisting of incomplete signals would cause great information loss while utilizing normal signal completion methods would contradict the hidden relationships between nodes.

Using the pulling process in monocrystalline silicone manufacturing as a motivating example, Figure 2 shows a diagram of the growth stove, which uses the Czochralski Method (CZ method), a commonly used technology in monocrystalline silicone manufacturing, for the pulling process. The whole system consists of multiple subsystems with different purposes such as vacuum control, gas supply, thermal field control, electricity supply, and transmission control systems. Monocrystalline silicone ingots are gradually formed when the crucible rotates, and a thread with a seed crystal attached to the end pulls upward from the molten polysilicon liquid. Due to the nature of monocrystalline silicone growth, the formation process consists of stages such as shoulder growth, body growth, and tail growth. Multiple sensors for measuring temperature, power, speed, weight, and length are installed in the system to collect data by monitoring and controlling the state of the process. As each process cycle usually lasts multiple days, the observations from each collected variable are shown in a functional form. These measured variables are physically connected with each other, and thus naturally constitute a complex relationship network that can be described by a DGM. Therefore, it is essential to learn the structure and estimate the relationship among the nodes of the DGM that represents the system. However, in harsh operating environments, some of the observations could be missing because of the reasons mentioned above. Poor contact of the sensors might cause continuous missing and high-temperature of internal environment might cause sensors generating invalid readings. Therefore, a novel method is

needed to accurately learn and monitor the structure of DGMs as well as functional relationships between nodes when using incomplete signals.

Consider the spatial-temporal modeling for the wind farm as another example, where a number of wind turbines are usually located in groups and a complex network system consists of various variables might influence power output of the whole wind farm (Pourhabib, Huang, and Ding 2016). However, wind farms are usually located in remote areas and long-distance data transmission would cause the poor quality of the available signals for variables like wind speed and pitch angles of turbine blades. As is shown in the above example, it is convinced that signal incompletion is an essential problem in modeling complex sensor systems like DGMs.

There have been many studies on the learning of DGMs, one of which used graphical LASSO to estimate the functional relationship between nodes of scalar observations (Yuan and Lin 2007). Although methods for structure learning and parameter estimation using DGMs with scalars have proven to be effective, they cannot be simply utilized in functional DGM learning. An extension of graphical LASSO has been proposed to address functional DGM learning (Qiao, Guo, and James 2019). Additionally, Bayesian and non-parametric methods have been shown to be effective (Zhu, Strawn, and Dunson 2016; Li and Solea 2018). These methods, however, are designed for undirected functional graphical models. To address the small sample size problem, a sparsity penalization approach has been developed to robustly learn the functional DGM (Sun, Huang, and Jin 2017). Group LASSO has also been utilized to effectively learn the structure together with the relevant parameters; this strategy employed a cyclic coordinate accelerated proximal gradient descent algorithm (Estrada, Paynabar, and Pacella 2021). However, the methods mentioned above are unable to address functional DGMs with incomplete signals, and therefore, an effective combination of the functional DGM learning method and signal completion method is needed.

Learning functional DGMs with incomplete signals can be considered an adjusted matrix completion problem, which is also a common problem in recommender systems (Matsuda and Komaki 2019). Among these matrix completion methods, matrix factorization (MF) has been shown to be effective under the low-rank assumption. There have been several variants of MF, including SVD Feature (Chen et al. 2012), Alternating Least Square (Takács and Tikk 2012) and Zipf Matrix Factorization (Wang 2021). However, these methods are not designed for incomplete problems with graphical structures. Alternatively, a simple solution executes these matrix completion methods for each node and then learns the structure and parameters as usual. This heuristic causes the results of matrix completion to violate the potential graphical structure since it neglects the functional relationships between nodes, which would be evident in the numerical experiments.

In general, there have been a few attempts to effectively combine signal completion methods and functional DGM learning methods. In this study, a novel adjusted matrix completion method is proposed to learn the structure and functional relationship of DGMs with incomplete signals. Based on the Maximum Margin Matrix Factorization (MMMF) algorithm, a specially designed penalty term has been added to the objective function of matrix completion to integrate the knowledge of DGMs. Specifically, when the structure of the DGM is unknown, several steps including signal completion, parameter estimation, and variable selection, are operated recurrently to obtain an accurate estimation of the parameters and structure of the DGM. Additionally, the proposed method is robust to different proportions of signal incompletion and various modes of missing data. In real-world cases, based on the learned DGM of the proposed method, a potential relationship between variables can be provided to help analyze the system network when little expert knowledge is available.

The remainder of this article is organized as follows: In Sec. 2, the proposed model and relevant parameter estimation procedures are presented in detail. Examples including numerical experiments and a real-world application are presented in Secs. 3 and 4, respectively, to verify the efficiency of the proposed method. Finally, conclusions and findings that can be drawn from this research are summarized in Sec. 5.

2. Model

For different application settings, the structural relationship may or may not to be known. Therefore, in this section, we first introduce the formulation of the problem and then develop learning methods for cases with known and unknown graph structures.

2.1. Problem formulation

Consider a system with M components, which serve as nodes of the DGM to be learned with N observations. $x_{mi}(t),\ m = 1, ..., M,\ i = 1, ..., N,\ t \in [0, T_m]$ denotes the ith observation of the mth node at time t. Note that we do not assume the lengths of signals from different nodes to be the same, which is suitable for cases where the sensors from different nodes might have different acquisition frequency of signals. Therefore, we assume the length of the signals from the mth node to be T_m. A simplified illustration of the data structure is shown in Figure 3, which depicts a simple graph with 1 root node and 2 leaf nodes.

In real-world applications, signals are observed at a specific frequency, that is, over a grid of size L_m for the mth node. Then, the collected data of observations from the DGM can be denoted as $\{x_{mit} | m = 1 \cdots M,\ i = 1 \cdots N,\ t = 1 \cdots L_m\}$. The data matrix from the mth node is denoted by $\boldsymbol{S}_m^{N \times L_m}$, and the element of the ith row and tth column are denoted as $x_{mit},\ i = 1 \cdots N,\ t = 1 \cdots L_m$. However, the signal matrix $\boldsymbol{S}_m$ is incomplete for various reasons, and the observed matrix is denoted as $\boldsymbol{S}_m^{\boldsymbol{\Omega}_m}$, where $\boldsymbol{\Omega}_m$ is the set of indices whose values are observed in the matrix. $\boldsymbol{\Omega}_m = \left\{(i,j) : j \in \Omega_m^i,\ i = 1, ..., N\right\}$, where $\Omega_m^i \subseteq \{1, ..., L_m\}$. Therefore, the final available dataset is $\left\{\boldsymbol{S}_1^{\boldsymbol{\Omega}_1}, \cdots, \boldsymbol{S}_M^{\boldsymbol{\Omega}_M}\right\}$.

The structure of the DGM is assumed to be a directed acyclic graph (DAG), which is a commonly used assumption in previous works (Sun, Huang, and Jin 2017; Estrada, Paynabar, and Pacella 2021). Therefore, the arcs of the DGM cannot fix a cycle, which allows the joint distribution of the nodes to be ordered. If there is an arc from i to j, then node i is called the parent of node j. Specifically, nodes without parents are called root nodes. A DAG can be uniquely expressed by a node set and an arc set, $\mathcal{G} = \{\mathcal{N}, \mathcal{A}\}$, where $\mathcal{N} = \{1, ..., M\}$, and $\mathcal{A}$ is a set of paired nodes $(i,j) \in \mathcal{A}$ if there exists an arc from i to j. In this work, the arcs are assumed to be a relationship in the form of a function-to-function linear regression as shown in Eq. (1):

$$x_{mi}(t) = \alpha_m(t) + \sum_{l \in Pa(m)} \int_0^{T_l} \gamma_{l,m}(s,t)\ x_{li}(s) ds + \varepsilon_{mi}(t) \tag{1}$$

where $Pa(m)$ is the parents' set of node m; $\gamma_{l,m}(s,t)$ is the F-to-F coefficient function that depicts the influence of the sth timepoint of node l on the tth timepoint of node m; and $\alpha_m(t)$ and $\varepsilon_{mi}(t)$

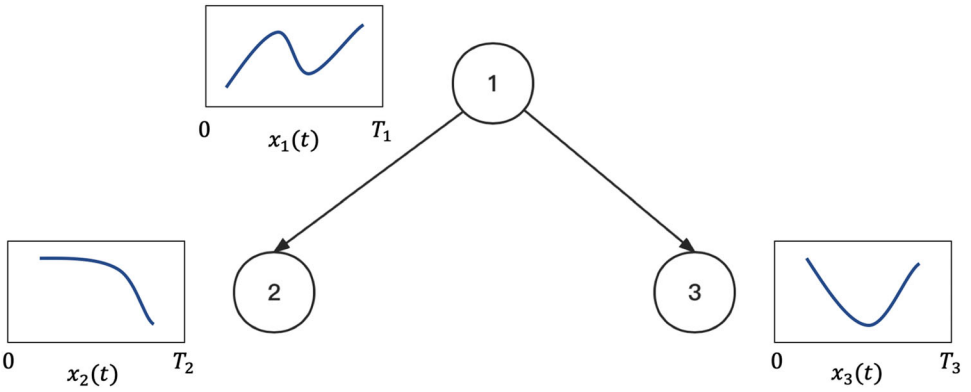

Figure 3. Illustrative example of the data structure.

are the constant term and the error term, respectively. Specifically, $\varepsilon_{mi}(t)$ is assumed to be the product of noise scale σ_m and a standard Gaussian random variable $\epsilon_{mi}(t)$, which makes the conditional distribution that shown in Eq. (2) given the DAG structure.

$$x_{mi}(t) \sim N\left(\alpha_m(t) + \sum\nolimits_{l \in Pa(m)} \int_0^{T_l} \gamma_{l,m}(s,t)\ x_{li}(s)ds, \sigma_m^2\right) \tag{2}$$

According to the description shown above, the aim is to learn the functional relationship $\gamma_{l,m}(s,t)\ , m = 1 \cdots M,\ l \in Pa(m)$ if the structure of the DGM is known, that is, if $\mathcal{A}$ is known (details are provided in Sec. 2.3) or to learn the functional relationship $\gamma_{l,m}(s,t)$ together with the potential structure when $\mathcal{A}$ is unknown (details are provided in Sec. 2.4) using the observed incomplete signal matrices $\left\{\boldsymbol{S}_1^{\boldsymbol{\Omega}_1}, \cdots, \boldsymbol{S}_M^{\boldsymbol{\Omega}_M}\right\}$.

2.2. Methodology framework

In this work, we separate the problem of DGM learning with incomplete signals into two scenarios: the graph structure $\mathcal{A}$ is (1) known or (2) unknown. Although it seems that these two scenarios are completely different, the methods used to solve them share many steps. The framework of the proposed method for addressing these two scenarios is shown in Figure 4.

When the structure of the graph is known, the whole graph learning procedure consists of three parts: MMMF, FPCR, and structure-based adjusted MMMF. The original MMMF is first utilized to complete the signal matrix of the aimed node only using information from its observed matrix, $\boldsymbol{S}_m^{\boldsymbol{\Omega}_m}$. Then, functional principal component regression (FPCR) is used to estimate the coefficients of the function-to-function regression (F-to-F regression), which is supported by the complete matrices of the target node and its parents. However, since the information from the F-to-F regression is not used in the former matrix completion step, an adjusted MMMF method is utilized to better complete the signal matrix by removing the influence of both the F-to-F regression effect and the low-rank effect from the partially observed signal matrix. These steps are executed recursively until certain convergence conditions are satisfied, with outputs of the estimated regression coefficients and complete matrices.

On the other hand, if the structure is unknown, most steps are the same as those for the known-structure scenario, except that when estimating the regression coefficients, group LASSO

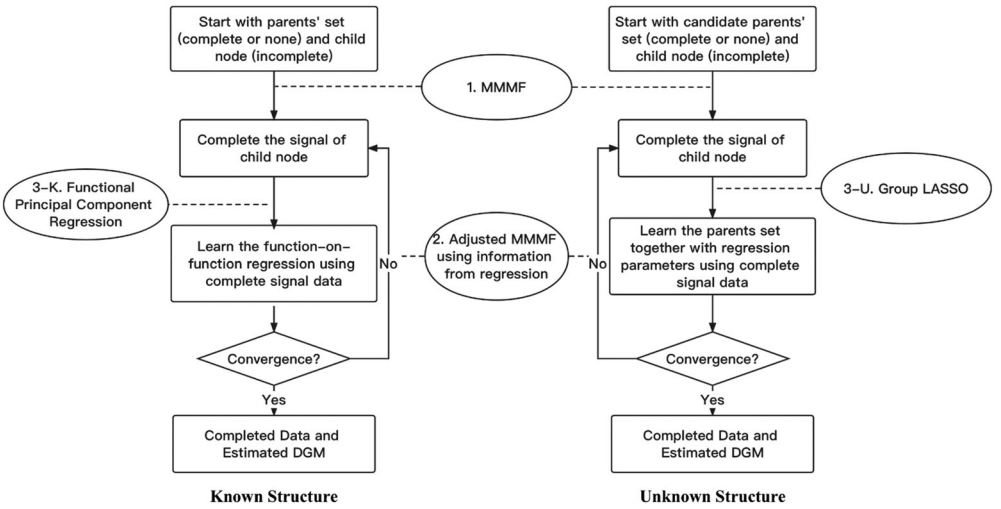

Figure 4. Flow diagram of the proposed method for each node in two scenarios (rectangles represent steps, and ellipses represent the used models).

is utilized to learn the parents from the candidate parents set of the target node and estimate the corresponding regression coefficients at the same time. Naturally, the output in this scenario contains learned structure $\mathcal{A}$ along with the F-to-F regression coefficients of each arc.

Additionally, there are nodes without parent nodes or candidate parent nodes, which we call root nodes. When completing the signal matrices using the root nodes, there is no regression effect from the parent nodes, and therefore, the only step involves using the MMMF once to obtain the estimated complete matrix. Note that the steps in the flowcharts should be operated on each node of the graph, and the input for the flowcharts is the incomplete matrix of the target node (child node) and complete matrices of its parents' node set, which are omitted if not available. Although the matrices from all nodes are assumed to be incomplete from the beginning, the conditions can be satisfied if the nodes are scanned and handled in a specific order, as suggested by the DAG assumption, because the output of the flowcharts contains a complete matrix of the child node that can be used to complete its child nodes.

In this framework, MMMF, FPCR, or group LASSO, and structure-based adjusted maximum margin matrix factorization (MMMF) are used in three steps recursively rather a joint framework. The main reason for this design is that it is difficult to give a explicit representation of structural information other than the regression estimation when considering dimensional reduction and matrix completion at the same time, since the completion results would affect the extracted subspace.

2.3. DGM learning with a known structure

In this section, steps from the known structure components in Figure 4 are shown in detail. In the first part, the input is a partially observed matrix, $\boldsymbol{S}_m^{\boldsymbol{\Omega}_m}$ for node m, and the output is a recovered complete matrix denoted by $\boldsymbol{S}_m^C$. In practice, since rows of $\boldsymbol{S}_m$ are usually observations of samples of a variable in the same process, the original matrix $\boldsymbol{S}_m$ is assumed to have a low-rank property. As for the completion technique, FPC is also widely used in data recovery (Zhou, Gebraeel, and Serban 2012; Sun, Liao, and Upadhyaya 2014). However, the subspace extracted by their methods is learned by a proportion of complete signals or given by specified basis functions like B-spline methods, which is not suitable when data missing exists in all signals with various missing patterns. MMMF is chosen as the method for matrix completion for the low-rank property with no constraints on the missing data, for example, missing proportions and missing patterns. The objective function of the MMMF is expressed as:

$$\begin{aligned} &\min_{\boldsymbol{S}_m^C} \|\boldsymbol{S}_m^{\boldsymbol{\Omega}_m} - (\boldsymbol{X}_m\boldsymbol{Y}_m)^{\boldsymbol{\Omega}_m}\|_F^2 + \lambda\left(\|\boldsymbol{X}_m\|_F^2 + \|\boldsymbol{Y}_m\|_F^2\right) \\ &s.t.\ \boldsymbol{S}_m^C = \boldsymbol{X}_m\boldsymbol{Y}_m \end{aligned} \tag{3}$$

where the recovered matrix $\boldsymbol{S}_m^C$ is set to the product of two low-rank matrices, $\boldsymbol{X}_m^{N\times K}$ and $\boldsymbol{Y}_m^{K\times L_m}(K \ll \min\{N, L_m\})$; $\|\cdot\|_F^2$ is the Frobenius norm; and λ is the tuning parameter, which can be selected by cross-validation together with rank K. Objective function (3) consists of two penalty terms. The first term causes the recovered matrix $\boldsymbol{S}_m^C$ to have small differences from $\boldsymbol{S}_m^{\boldsymbol{\Omega}_m}$ in the observed indices, and the second term guarantees that recovered matrix $\boldsymbol{S}_m^C$ has a smaller rank. To solve this optimization problem, there are many existing algorithms (Balzano, Nowak, and Recht 2010; Toh and Yun 2010). One straightforward method is the gradient descent algorithm. After initialization of $\boldsymbol{X}_m$ and $\boldsymbol{Y}_m$, $\boldsymbol{X}_m = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N)^T$, $\boldsymbol{Y} = (\boldsymbol{y}_1, ..., \boldsymbol{y}_{L_m})$ are set, and $\boldsymbol{x}_i$, $\boldsymbol{y}_j (i = 1\cdots N,\ j = 1\cdots L_m)$ are K-dimensional column vectors. The loss function of Eq. (3) can be rewritten as:

$$L = \sum\nolimits_{(i,j)\in\boldsymbol{\Omega}_m}\left(s_{ij} - \boldsymbol{x}_i^T\boldsymbol{y}_j\right)^2 + \lambda\left(\|\boldsymbol{X}_m\|_F^2 + \|\boldsymbol{Y}_m\|_F^2\right) \tag{4}$$

where s_{ij} is the element of the ith row and jth column in $\mathbf{S}_m^{\mathbf{\Omega}_m}$. The following updating equations are recursively executed for each $(i,j) \in \mathbf{\Omega}_m$ until convergence.

$$\boldsymbol{x}_i^{(t+1)} = \boldsymbol{x}_i^{(t)} + \kappa\left(\delta_{ij}\boldsymbol{y}_j^{(t)} - \lambda\boldsymbol{x}_i^{(t)}\right) \tag{5}$$

$$\boldsymbol{y}_j^{(t+1)} = \boldsymbol{y}_j^{(t)} + \kappa\left(\delta_{ij}\boldsymbol{x}_i^{(t)} - \lambda\boldsymbol{y}_j^{(t)}\right) \tag{6}$$

where $\delta_{ij} = s_{ij} - \boldsymbol{x}_i^T\boldsymbol{y}_j$ and κ is the learning rate. The convexity of the optimization problem (3) makes the algorithm efficient for getting a better solution in an iterative way.

In the second part, the complete matrices of the target node and its parent nodes are used to estimate the coefficients of the F-to-F regression. Additionally, an estimated matrix for the target node can be provided according to the resulting regression coefficients. To better explain this part, theoretical derivations are provided before the discretized form is shown. Since the dimensions of the signal matrices, L_m and N, are relatively large, certain dimension reduction methods are needed. In this work, functional principal component analysis (FPCA) is chosen; if domain knowledge is available, prespecified basis functions could also be utilized (Estrada, Paynabar, and Pacella 2021). Recall that for the functional signal collected from node m, $x_{mi}(t)$ can be decomposed by eigenfunctions given by the covariance function $cov\big(x_m(s), x_m(t)\big)$ as follows:

$$x_{mi}(t) = \sum_{p=1}^{\infty}\zeta_{mip}\psi_{mp}(t) \tag{7}$$

where $\psi_{mp}(t)$ are eigenfunctions and ζ_{mip}s represents the respective FPC scores. Similarly, the functional signals from the parents of node m can be decomposed as:

$$x_{li}(t) = \sum_{q=1}^{\infty}\xi_{liq}\theta_{lq}(t), \;\; l \in Pa(m) \tag{8}$$

where $\theta_{lq}(t)$ are eigenfunctions and ξ_{liq}s are the respective FPC scores. Additionally, the coefficient function $\gamma_{l,m}(s,t)$ in Eq. (1) can also be decomposed by eigen functions $\psi_{mp}(t)$ and $\theta_{lq}(t)$ as in (Horváth and Horváth 2012):

$$\gamma_{l,m}(s,t) \;\; = \sum_{p=1}^{\infty}\sum_{q=1}^{\infty}b_{mlpq}\psi_{mp}(t)\theta_{lq}(s) \tag{9}$$

Substituting Eqs. (7) and (8) into Eq. (1), the function-to-function linear regression can be rewritten as:

$$\sum_{p=1}^{\infty}\zeta_{mip}\psi_{mp}(t) = \sum_{l\in Pa(m)}\int_0^{T_l}\sum_{p=1}^{\infty}\sum_{q=1}^{\infty}b_{mlq}\psi_{mp}(t)\theta_{lq}(s)\sum_{q=1}^{\infty}\xi_{liq}\theta_{lq}(s)ds + \varepsilon_{mi}(t) \tag{10}$$

To achieve the objective of dimensional reduction, we select the former P_m and P_l eigenfunctions as approximation of the original functional signals $x_{mi}(t)$ and $x_{li}(t)$, respectively. Together with the orthogonal properties of $\psi_{mp}(t)$s and $\theta_{lq}(t)$ s, Eq. (10) can be reduced as:

$$\sum_{p=1}^{P_m}\zeta_{mip}\psi_{mp}(t) = \sum\nolimits_{l\in Pa(m)}\sum_{p=1}^{P_m}\sum_{q=1}^{P_l}b_{mlpq}\xi_{liq}\psi_{mp}(t) + \sigma_m\epsilon_{mi}(t) \tag{11}$$

Furthermore, multiplying Eq. (11) by $\psi_{mp}(t)$ obtains a multiple linear regression form with FPC scores serving as variables:

$$\zeta_{mip} = \sum_{l \in Pa(m)} \sum_{q=1}^{P_l} b_{mlpq} \xi_{liq} + \sigma_m \epsilon_{mip} \tag{12}$$

where $\epsilon_{mip} = \int_0^{T_m} \psi_{mp}(t) \epsilon_{mi}(t) dt$. The matrix form of Eq. (12) is expressed as follows:

$$\boldsymbol{\zeta}_m = \sum_{l \in Pa(m)} \boldsymbol{\xi}_l \boldsymbol{b}_{ml} + \sigma_m \mathrm{E}_m \tag{13}$$

where $\boldsymbol{\zeta}_m \in \mathbb{R}^{N \times P_m}$ with the element in the ith row and pth column are ζ_{mip}, $i = 1 \cdots N$, $p = 1 \cdots P_m$; $\boldsymbol{\xi}_l \in \mathbb{R}^{N \times P_l}$ with the element in the ith row and qth column are ξ_{liq}, $i = 1 \cdots N$, $q = 1 \cdots P_l$; $\boldsymbol{b}_{ml} \in \mathbb{R}^{P_l \times P_m}$ is the coefficient matrix where the element in the qth row and pth column is b_{mlpq}, $q = 1 \cdots p_l$, $p = 1 \cdots P_m$; and E_m is the error matrix. This form can be developed further by integrating the variables from the parent nodes and setting $\boldsymbol{\Xi}_m \in \mathbb{R}^{N \times \sum_{l \in Pa(m)} P_l}$ to be the joint FPC score matrix and $\boldsymbol{B}_m \in \mathbb{R}^{\sum_{l \in Pa(m)} P_l \times P_m}$ to be the joint coefficient matrix, which would obtain:

$$\boldsymbol{\zeta}_m = \boldsymbol{\Xi}_m \boldsymbol{B}_m + \sigma_m \mathrm{E}_m \tag{14}$$

and the estimation of coefficient matrix $\boldsymbol{B}_m$ is provided by the squared loss:

$$\hat{\boldsymbol{B}}_m = \left(\boldsymbol{\Xi}_m^T \boldsymbol{\Xi}_m\right)^{-1} \boldsymbol{\Xi}_m \boldsymbol{\zeta}_m \tag{15}$$

In real applications with discretized signal matrices $\boldsymbol{S}_m^C$ and $\boldsymbol{S}_l^C$ s, PCs $\boldsymbol{\zeta}_m$ and $\boldsymbol{\Xi}_m$ are similarly extracted, and the SVD result of signal matrix from the target node is $\boldsymbol{S}_m^C = \boldsymbol{U}_m \boldsymbol{\Lambda}_m \boldsymbol{V}_m^T$. Then, the prediction matrix for the incomplete signal matrix of the target node is as follows:

$$\boldsymbol{S}_m^R = \boldsymbol{\Xi}_m \hat{\boldsymbol{B}}_m \cdot \tilde{\boldsymbol{V}}_m = \hat{\boldsymbol{\zeta}}_m \tilde{\boldsymbol{V}}_m \tag{16}$$

where $\tilde{\boldsymbol{V}}_m \in \mathbb{R}^{P_m \times L_m}$ is the transpose of the former P_m columns of $\boldsymbol{V}_m^T$, which also represents the projection matrix of $\boldsymbol{S}_m^C$; and $\boldsymbol{S}_m^R$ is the output of the second part, which is the estimated signal matrix from the F-to-F linear regression representing the information from the graph structure. Recall that the input to the second part $\boldsymbol{S}_m^C$ is a low-rank matrix, that is, $rank\left(\boldsymbol{S}_m^C\right) = K$. If the number of principal components is selected to be K, then there is no information loss from the original matrix $\boldsymbol{S}_m^C$ to the score matrix $\boldsymbol{\zeta}_m$.

In the third part, a more accurate recovered matrix $\boldsymbol{S}_m^C$ is estimated by an adjusted MMMF method using information from both the incomplete signal matrix and the graph structure, that is, $\boldsymbol{S}_m^{\boldsymbol{\Omega}_m}$, $\boldsymbol{S}_m^R$, which is also the main contribution of this work. The objective function of the adjusted MMMF is as follows:

$$\begin{gathered} \min_{\boldsymbol{S}_m^C} \|\boldsymbol{S}_m^{\boldsymbol{\Omega}_m} - (\boldsymbol{X}_m \boldsymbol{Y}_m)^{\boldsymbol{\Omega}_m}\|_F^2 + \mu \|\boldsymbol{S}_m^{R\,\bar{\boldsymbol{\Omega}}_m} - (\boldsymbol{X}_m \boldsymbol{Y}_m)^{\bar{\boldsymbol{\Omega}}_m}\|_F^2 + \lambda\left(\|\boldsymbol{X}_m\|_F^2 + \|\boldsymbol{Y}_m\|_F^2\right) \\ s.t.\ \boldsymbol{S}_m^C = \boldsymbol{X}_m \boldsymbol{Y}_m \end{gathered} \tag{17}$$

where $\bar{\boldsymbol{\Omega}}_m = \left\{(i,j) | (i,j) \notin \boldsymbol{\Omega}_m, i = 1 \cdots N, j = 1 \cdots L_m\right\}$ is the complementary set; $\boldsymbol{S}_m^{R\,\bar{\boldsymbol{\Omega}}_m}$, $(\boldsymbol{X}_m \boldsymbol{Y}_m)^{\bar{\boldsymbol{\Omega}}_m}$ are the elements from the indices in $\bar{\boldsymbol{\Omega}}_m$ from the original matrices $\boldsymbol{S}_m^R$ and $(\boldsymbol{X}_m \boldsymbol{Y}_m)$, respectively; and the other notations have the same meanings as in Eq. (3). Compared with the original objective function of the MMMF, a specially designed penalty (the second term in Eq. (17)) is added, which comes from the straightforward idea that the final recovered signal matrix should not violate the partially observed matrix $\boldsymbol{S}_m^{\boldsymbol{\Omega}_m}$, or the estimation from the F-to-F regression relationship $\boldsymbol{S}_m^R$. Note that $\boldsymbol{S}_m^R$ comes from the signal matrices from parent nodes after completion and the estimated coefficient functions. Since the completion process is operated from root nodes to seed nodes, the signal matrices from parent nodes of node m, that is, $\left\{\boldsymbol{S}_j \,|i \in Pa(m)\ Cpa(m)\right\}$

are considered to be relatively accurate. On the other hand, we have assumed that the relationship between nodes is modeled in the form of linear F-to-F regression. Therefore, the prediction matrix $\boldsymbol{S}_m^R$ contains the structural influence from parents' nodes. Although it might not be accurate, it is considered to be a better prediction on the unavailable indices $\bar{\boldsymbol{\Omega}}_m$ compared to the complete matrix from former iterations. This is also the reason why we need to recursively operate the estimation steps A tuning parameter μ is also added to balance these two effects to fit various situations. Similar to the first part, the loss function and related updating equations are expressed as follows:

$$L = \sum_{(i,j)\in\boldsymbol{\Omega}_m}\left(s_{ij}-\boldsymbol{x}_i^T\boldsymbol{y}_j\right)^2 + \mu\sum_{(i,j)\in\bar{\boldsymbol{\Omega}}_m}\left(s_{ij}^R-\boldsymbol{x}_i^T\boldsymbol{y}_j\right)^2 + \lambda\left(\|\boldsymbol{X}_m\|_F^2+\|\boldsymbol{Y}_m\|_F^2\right) \tag{18}$$

$$\boldsymbol{x}_i^{(t+1)} = \boldsymbol{x}_i^{(t)} + \kappa\left(\delta_{ij}\boldsymbol{y}_j^{(t)} - \lambda\boldsymbol{x}_i^{(t)}\right) \tag{19}$$

$$\boldsymbol{y}_j^{(t+1)} = \boldsymbol{y}_j^{(t)} + \kappa\left(\delta_{ij}\boldsymbol{x}_i^{(t)} - \lambda\boldsymbol{y}_j^{(t)}\right) \tag{20}$$

$$\delta_{ij} = \begin{cases} s_{ij}-\boldsymbol{x}_i^T\boldsymbol{y}_j, & if\ \ (i,j)\in\boldsymbol{\Omega}_m \\ \mu\left(s_{ij}^R-\boldsymbol{x}_i^T\boldsymbol{y}_j\right), & otherwise \end{cases} \tag{21}$$

where $\boldsymbol{X}_m = (\boldsymbol{x}_1,\ \cdots,\boldsymbol{x}_N)^T$, $\boldsymbol{Y}_m = (\boldsymbol{y}_1,...,\boldsymbol{y}_{L_m})$; and κ is the learning rate. To fasten the convergence speed and strengthen the stability of the estimation result, the solution from last recurrent iteration, that is, $\boldsymbol{S}_m^{C\,(\tau-1)}$, serves as the initial solution. Specially, the solution of Eq. (4) actually provides the initial solution in the first recurrent step.

In general, we start with the first part and operate the last two parts recursively until convergence. There are two parts in the proposed framework that involves the convergence condition issues. The first part is when using stochastic gradient descent algorithm to solve MMMF and adjusted MMMF, that is, Eqs. (4) and (18). In this work, the convergence condition for these two solutions is given as follows:

$$\|\boldsymbol{X}_m^{(t)}-\boldsymbol{X}_m^{(t-1)}\|_F^2 + \|\boldsymbol{Y}_{\mathrm{m}}^{(t)}-\boldsymbol{Y}_{\mathrm{m}}^{(t-1)}\|_F^2 < h_0 \tag{22}$$

where t is the iteration number of the algorithm; h_0 is a predetermined threshold, which is set according to the dimensions of $\boldsymbol{X}_m$ and $\boldsymbol{Y}_m$. The second part is when judging whether to stop the recurrent steps shown in Figure 4, that is, FPCR + adjusted MMMF. The convergence condition is given as:

$$\|\boldsymbol{S}_m^{C\,(\tau)}-\boldsymbol{S}_m^{C\,(\tau-1)}\|_F^2 < h_1 \tag{23}$$

where τ is the iteration number; h_1 is a predetermined threshold, which is set according to the dimensions of $\boldsymbol{S}_m^C$. The proposed method with known structure is summarized in Algorithm 1.

Algorithm 1: Matrix completion with known graphical structure

Input: incomplete signal matrices $\{\mathbf{S}_{\mathrm{m}}^{\boldsymbol{\Omega}_{\mathrm{m}}}\}$, $m\in\{1,...,M\}$; structural information $Pa(m)$, $m\in\{1,...,M\}$

Output: complete signal matrices $\{\mathbf{S}_{\mathrm{m}}^{c}\}$, $m\in\{1,...,M\}$

(1) Denote the node set $\mathcal{M}_r$ to be the nodes that are ready for completion, $\mathcal{M}_u$ to the nodes that are not ready for completion, $\mathcal{M}_c$ to the nodes that are completed and the whole node set $\mathcal{M}=\{1,...,M\}$

(2) Initialize $\mathcal{M}_r = \{k \in \mathcal{M} | Pa(k) = \emptyset\}$, $\mathcal{M}_u = \{k \in \mathcal{M} | Pa(k) = \emptyset\}$, $\mathcal{M}_c = \emptyset$
(3) **while** $\mathcal{M}_c \neq \mathcal{M}$ **do**
(4) Randomly choose a node, m, from $\mathcal{M}_r$
(5) Utilize MMMF in (3)–(6) based on $\mathbf{S}_{\mathrm{m}}^{\mathbf{\Omega}_{\mathrm{m}}}$ to get $\mathbf{S}_{\mathrm{m}}^{c\ (0)}$
(6) **if** $Pa(m) = \emptyset$ **then**
(7) $\mathbf{S}_{\mathrm{m}}^{c} \leftarrow \mathbf{S}_{\mathrm{m}}^{c\ (0)}$
(8) **else**
(9) Set $\tau = 0$
(10) **do**
(11) Utilize FPCR (11)–(16) based on $\mathbf{S}_{\mathrm{m}}^{c\ (\tau)}$ and $\left\{\mathbf{S}_j^c | j \in Pa(m)\right\}$ to get $\mathbf{S}_{\mathrm{m}}^{R\ (\tau)}$
(12) Utilize the adjust MMMF in (17)–(21) based on $\mathbf{S}_{\mathrm{m}}^{\mathbf{\Omega}_{\mathrm{m}}}, \mathbf{S}_{\mathrm{m}}^{R\ (\tau)}$ to get $\mathbf{S}_{\mathrm{m}}^{c\ (\tau+1)}$
(13) Set $\tau \leftarrow \tau + 1$
(14) **until** convergence condition (23) is satisfied
(15) $\mathbf{S}_{\mathrm{m}}^{c} \leftarrow \mathbf{S}_{\mathrm{m}}^{c\ (\tau)}$
(16) **end do**
(17) **end if**
(18) Update $\mathcal{M}_c \leftarrow \mathcal{M}_c \cup \{m\}$
(19) Update $\mathcal{M}_r \leftarrow \{k \in \mathcal{M} \backslash \mathcal{M}_c Pa(k) = \emptyset\} \cup \{k \in \mathcal{M} \backslash \mathcal{M}_c Pa(k) \subseteq \mathcal{M}_c\}$
(20) Update $\mathcal{M}_u \leftarrow \mathcal{M} \backslash \{\mathcal{M}_c \cup \mathcal{M}_r\}$
(21) **end while**
(22) **return** $\{\mathbf{S}_{\mathrm{m}}^{c}\}$, $m \in \{1, \ldots, M\}$

2.4. DGM learning with an unknown structure

Different from Sec. 2.3, it is assumed that the graph structure $\mathcal{A}$ is unknown in this scenario. However, a new assumption is added: that the candidate parent set of each node $Cpa(m)$, $m \in \mathcal{N}$ is known and is an empty set when the mth node is a root node. Naturally, the real parent set is a subset of the candidate parent set, that is, $Pa(m) \subseteq Cpa(m)$. This assumption is reasonable in real-world applications. Even if no relative domain knowledge is known, we can set $Cpa(m) = \{1 \cdots m-1\}$, $m \in \mathcal{N}$ to maintain the DAG assumption. Comparing these steps to those when the structure of the graph is known, the only difference is the second part, while the other two parts are the same as in Sec. 2.3, which will not be explained in this section.

In the second part of this scenario, an estimation of the signal matrix is also needed in the case of an unknown structure, which dictates that the input of this part is $\boldsymbol{S}_m^C$ and $\boldsymbol{S}_l^C$, $l \in Cpa(m)$. Similarly, PCs are extracted to construct the linear regression function as:

$$\boldsymbol{\zeta}_m = \sum_{l \in Cpa(m)} \boldsymbol{\xi}_l \boldsymbol{b}_{ml} + \sigma_m \mathrm{E}_m \tag{24}$$

where the notations have the same meanings in Eq. (14). However, the output of this part contains not only the parameters $\boldsymbol{b}_{ml}$ and resulting estimated matrix $\boldsymbol{S}_m^R$, but also the learned structure, that is, $\widehat{Pa(m)}$, which means that $\hat{\boldsymbol{b}}_{ml} = 0$ when $l \notin \widehat{Pa(m)}$. Therefore, the group LASSO method is utilized with the respective loss function:

$$\begin{aligned} L(\boldsymbol{b}_{ml} l \in Cpa(m)) = & \frac{1}{2} \left\| \boldsymbol{\zeta}_m - \sum_{l \in Cpa(m)} \boldsymbol{\xi}_l \boldsymbol{b}_{ml} \right\|_2^2 \\ & + \mu \sum_{l \in Cpa(m)} \sqrt{q_{ml}} \|\boldsymbol{b}_{ml}\|_2 + \frac{\lambda}{2} \sum_{l \in Cpa(m)} \|\boldsymbol{b}_{ml}\|_2^2 \end{aligned} \tag{25}$$

where q_{ml} is the size of $\boldsymbol{b}_{ml}$ and μ, λ are relative tuning parameters. The first term is the squared loss from Eq. (24). The second penalty term ensures the sparsity of the coefficient matrix, where

the first-order penalty selects variables that are highly correlated with the output variables and omits the rest (Hastie, Tibshirani, and Wainwright 2015). The third penalty term selects a few variables from groups that are highly correlated with others (Zou and Hastie 2005). The details of the algorithm in terms of minimizing this loss function and obtaining estimated coefficients $\hat{\boldsymbol{b}}_{ml}$ can be found in a previous work (Estrada, Paynabar, and Pacella 2021). Therefore, the estimated signal matrix from the F-to-F regression effect $\boldsymbol{S}_m^R$ can be calculated similarly to Sec. 2.3, and $\widehat{Pa(m)}$ is also achieved according to $\hat{\boldsymbol{b}}_{ml}$, that is, $\widehat{Pa(m)} = \left\{ l \in Cpa(m) | \hat{\boldsymbol{b}}_{ml} \neq 0 \right\}$.

In this scenario, the convergence condition for the recurrent steps, that is, group LASSO + adjusted MMMF is given as:

$$\|\boldsymbol{S}_m^{C\,(\tau)} - \boldsymbol{S}_m^{C\,(\tau-1)}\|_F^2 < h_1 \;\; and \;\; \widehat{Pa(m)}^{(\tau)} = \widehat{Pa(m)}^{(\tau-1)} \tag{26}$$

The proposed method with unknown structure is summarized in Algorithm 2.

Algorithm 2: Matrix completion with unknown graphical structure

Input: incomplete signal matrices $\left\{\mathbf{S}_{\mathrm{m}}^{\boldsymbol{\Omega}_{\mathrm{m}}}\right\}$, $m \in \{1, ..., M\}$; inaccurate structural information $Cpa(m)$, $m \in \{1, ..., M\} = \mathcal{M}$

Output: complete signal matrices $\{\mathbf{S}_{\mathrm{m}}^{c}\}$, $m \in \{1, ..., M\}$ and estimated structure $\widehat{Pa(m)}$, $m \in \{1, ..., M\}$

(1) Denote the node set $\mathcal{M}_r$ to be the nodes that are ready for completion, $\mathcal{M}_u$ to the nodes that are not ready for completion, $\mathcal{M}_c$ to the nodes that are completed and the whole node set $\mathcal{M} = \{1, ..., M\}$
(2) Initialize $\mathcal{M}_r = \left\{k \in \mathcal{M} | Cpa(k) = \emptyset\right\}$, $\mathcal{M}_u = \left\{k \in \mathcal{M} | Cpa(k) = \emptyset\right\}$, $\mathcal{M}_c = \emptyset$
(3) while $\mathcal{M}_c \neq \mathcal{M}$ do
(4) Randomly choose a node, m, from $\mathcal{M}_r$
(5) Utilize MMMF in (3)–(6) based on $\mathbf{S}_{\mathrm{m}}^{\boldsymbol{\Omega}_{\mathrm{m}}}$ to get $\mathbf{S}_{\mathrm{m}}^{c\,(0)}$
(6) if $Cpa(m) = \emptyset$ then
(7) $\mathbf{S}_{\mathrm{m}}^{c} \leftarrow \mathbf{S}_{\mathrm{m}}^{c\,(0)}$, $\widehat{Pa(m)} \leftarrow \emptyset$
(8) else
(9) Set $\tau = 0$ and $\widehat{Pa(m)}^{(0)} = Cpa(m)$
(10) do
(11) Utilize group LASSO (24)-(25) based on $\mathbf{S}_{\mathrm{m}}^{c\,(\tau)}$, $\left\{\mathbf{S}_j^c j \in \widehat{Pa(m)}^{(\tau)}\right\}$
(12) to get $\mathbf{S}_{\mathrm{m}}^{R\,(\tau)}$ and $\widehat{Pa(m)}^{(\tau+1)}$
(13) Utilize the adjust MMMF in (17)–(21) based on $\mathbf{S}_{\mathrm{m}}^{\boldsymbol{\Omega}_{\mathrm{m}}}, \mathbf{S}_{\mathrm{m}}^{R\,(\tau)}$ to get $\mathbf{S}_{\mathrm{m}}^{c\,(\tau+1)}$
(14) Set $\tau \leftarrow \tau + 1$
(15) until convergence condition (26) is satisfied
(16) $\mathbf{S}_{\mathrm{m}}^{c} \leftarrow \mathbf{S}_{\mathrm{m}}^{c\,(\tau)}$, $\widehat{Pa(m)} \leftarrow \widehat{Pa(m)}^{(\tau)}$
(17) end do
(18) end if
(19) Update $\mathcal{M}_c \leftarrow \mathcal{M}_c \cup \{m\}$
(20) Update $\mathcal{M}_r \leftarrow \left\{k \in \mathcal{M} \backslash \mathcal{M}_c Cpa(k) = \emptyset\right\} \cup \left\{k \in \mathcal{M} \backslash \mathcal{M}_c Cpa(k) \subseteq \mathcal{M}_c\right\}$
(21) Update $\mathcal{M}_u \leftarrow \mathcal{M} \backslash \{\mathcal{M}_c \cup \mathcal{M}_r\}$
(22) end while
(23)return $\{\mathbf{S}_{\mathrm{m}}^{c}\}$ $\left\{\widehat{Pa(m)}\right\}$, $m \in \{1, ..., M\}$

2.5. Selection of relevant parameters

In the proposed method, there are many parameters to be tuned; recall that we can set the number of PCs extracted A and the rank of matrices $\boldsymbol{X}_m$ and $\boldsymbol{Y}_m$, K to be equal ($A = K = 10$ in the simulation studies) to avoid information loss in the regression estimation part. In real-world

applications, A can be selected from PCA with complete signal matrices for each node. On the other hand, the criterion function for selecting penalty parameters (λ, μ) is expressed as:

$$L_m(\lambda, \mu) = \frac{1}{N_{valid}} \sum_{i=1}^{N_{valid}} \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} (x_{mi}(t) - \hat{x}_{mi}(t))^2 \tag{27}$$

where $\hat{x}_{mi}(t)$ results from $\boldsymbol{S}_m^C$ for root nodes and $\hat{x}_{mi}(t) = \sum_{l \in Pa(m)} \int_0^{T_l} \hat{\gamma}_{lm}(s,t) x_{li}(s) ds$ for other nodes; N_{valid} is the number of validation samples; and $\hat{\gamma}_{lm}(s,t)$ is the coefficient function estimated by the model with parameters (λ, μ). The candidate sets for the parameters in this work are $C_\lambda = \{10^{-9}, 10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}\}$ and $C_\mu = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$. In real-world cases, the selection of C_λ can be adjusted by the size of $|\mathcal{N}|$ and dimensions of signal matrices. Similar to the parameter selection criterion, the prediction accuracy is measured as:

$$MSPE = \frac{1}{|\mathcal{N}|} \sum_{m \in \mathcal{N}} \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} (x_{mi}(t) - \hat{x}_{mi}(t))^2 \tag{28}$$

where N_{test} is the number of testing samples.

3. Numerical simulations

In this section, numerical experiments are conducted to verify the efficiency of the proposed DGM learning framework. As the comparison method, the MMMF is utilized separately on each node to complete the signal matrices, and then the original DGM learning method with complete signals is executed, which is named as signal completion plus DGM learning (SC-DGML). The proposed method illustrated in Sec. 2 is named graph-based signal completion (GBSC). The advantages of the proposed method in the following simulation studies are shown in terms of three aspects: (1) different scales of error in signals; (2) different missing data patterns; and (3) different graph structures, including the density of edges and number of nodes.

Similar to a previous work (Estrada, Paynabar, and Pacella 2021), functional signals in the simulation studies were generated as follows: for the ith sample of the root node r, it is designed to be combinations of three Gaussian processes:

$$x_{ri}(t) = gp_{r1}(t) + \sigma_r \times \left(gp_{r2i}(t) + gp_{r3i}(t)\right) \tag{29}$$

where the covariance functions of $gp_{r1}(t)$, $gp_{r2i}(t)$, and $gp_{r3i}(t)$ are $\Sigma_1(t,t') = e^{-10\left(t-t'\right)^2}$, $\Sigma_2(t,t') = e^{-0.1\left(t-t'\right)^2}$, and $\Sigma_3(t,t') = 1$, respectively. $gp_{r1}(t)$ serves as the curve trend of node r, $gp_{r2i}(t)$ is the autocorrelated effect function in each observation and $gp_{r3i}(t)$ is the white noise function. For the observations from other nodes, the F-to-F linear regression function is designed to fit the assumption in Sec. 2:

$$x_{mi}(t) = \sum\nolimits_{l \in Pa(m)} \int_0^{T_l} \gamma_{lm}(s,t)\ x_{li}(s) ds + \sigma_m \epsilon_{mi}(t) \tag{30}$$

where the coefficient function is designed as $\gamma_{lm}(s,t) = \sum_{k=1}^{3} a_{klm}(t) b_{klm}(s)$; $a_{klm}(t), b_{klm}(s)$ are GPs with Σ_1, $k = 1, 2, 3$; $\epsilon_{mi}(t)$ is a GP with Σ_3, that is, standard normally distributed variables; and σ_r, σ_m are the respective scales of error. For simplicity, the number of grid points is set equal among all nodes, that is, $L_m = L = 50$, and $T_m = T = 1$, which means the observed timepoints are $t \in \mathcal{T} = \{0.02, 0.04, ..., 1\}$.

The generated data are divided into three sets: a training set, validation set and test set, for which $N_{train} = 80, N_{valid} = 20$ and $N_{test} = 50$. Additionally, the, data matrices from the training set are incomplete, while the validation and test matrices are complete. The steps for the whole simulation are as follows: first, we generate 100, that is, $N_{train} + N_{valid}$ complete signals and

randomly choose N_{train} samples to execute random deletion; second, we execute the methods on the N_{train} incomplete signals to construct a trained model using different combinations of parameters, and N_{valid} samples are used to test the efficiency of the trained model for the selection of parameters; and finally, N_{test} complete samples are tested in terms of the prediction accuracy performance of the methods using the learned DGM and the selected parameters.

To evaluate the performances of the two methods in different situations, four graphs denoted as G1, G2, G3, and G4 are designed to simulate DGMs with a large/small number of nodes and dense/sparse arcs. The detailed settings and structures of the designed graphs are shown in Figure 5 and Table 1.

In the first part, it is assumed that the structure is known, and the influence of the graphs, including the number of nodes and the density of arcs, is studied. The missing proportion of data is set to $p_{miss} = 0.5$, which means we randomly choose $\mathcal{T}_{miss} \subset \mathcal{T}(|\mathcal{T}_{miss}| = p_{miss} \times |\mathcal{T}|)$ and delete $x_{mi}(t),\ t \in \mathcal{T}_{miss}$ for each signal of each node. The scale of error is also considered, that is, $\sigma_m = \sigma = 0.1$ (low noise) and 0.5 (high noise). The results for the two methods in terms of

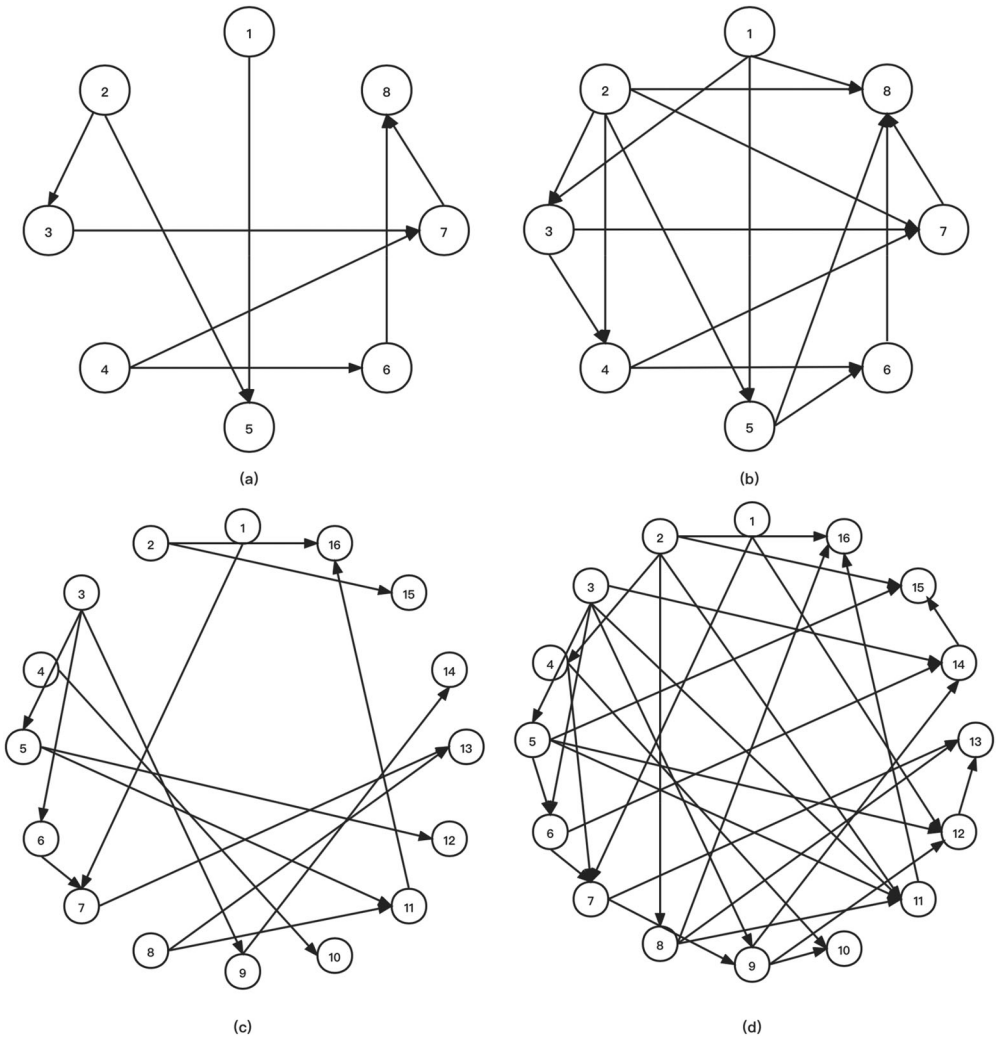

Figure 5. The four designed graphs. (a) G1: 8 nodes and 8 arcs (1/8 density); (b) G2: 8 nodes and 16 arcs (1/4 density); (c) G3: 16 nodes and 16 arcs (1/16 density); and (d) G4: 16 nodes and 32 arcs (1/8 density).

Table 1. Settings of relevant graphs in the following simulation studies.

	Number of nodes	Number of edges	Density of edges
G1	8	8	1/8
G2	8	16	1/4
G3	16	16	1/16
G4	16	32	1/8

Table 2. Prediction accuracy of the two methods for G1–G4 under two scales of error when the structure is known and $p_{miss} = 0.5$ (standard errors are shown in the parenthesis).

	Scale of error	GBSC	SC-DGML
G1	0.1	0.617(0.20)	1.178(1.26)
	0.5	0.732(0.21)	3.341(4.28)
G2	0.1	0.649(0.30)	0.873(0.56)
	0.5	0.840(0.59)	4.054(3.46)
G3	0.1	0.711(0.36)	1.278(1.35)
	0.5	1.165(1.19)	19.668(43.21)
G4	0.1	1.739(1.85)	3.143(4.86)
	0.5	2.382(3.79)	37.654(94.38)

Table 3. Prediction accuracy of the two methods with different missing proportions of data under three missing modes when the structure is known (standard errors are shown in the parenthesis).

		Missing proportions			
Missing modes	Methods	0	0.25	0.5	0.75
Random missing	GBSC	0.449(0.07)	0.519(0.15)	0.617(0.20)	0.804(0.40)
	SC-DGML	0.450(0.06)	0.611(0.22)	1.178(1.26)	16.253(21.67)
Imbalanced missing	GBSC	0.449(0.07)	0.503(0.10)	0.534(0.16)	0.818(0.34)
	SC-DGML	0.450(0.06)	0.509(0.11)	0.745(0.48)	24.752(37.41)
Non-uniform missing	GBSC	0.449(0.07)	0.501(0.37)	0.587(0.29)	0.809(0.49)
	SC-DGML	0.450(0.06)	0.541(0.25)	1.181(1.72)	22.715(41.33)

learning these four graphs with incomplete signals are shown in Table 2, with $N_{repli} = 100$ replication times.

Several findings could be extracted from the results. Naturally, prediction errors increase when the scale of error increases and when the structure of graph becomes complicated, that is, the number of nodes or arcs increases. Compared to the effect from denser arcs, the increase in the number of nodes would make the prediction error grow faster. Additionally, the proposed GBSC method is better than SC-DGML in all situations because that GBSC combines the signal combination step and coefficient estimation step and integrates information from the graph structure when completing the signal matrices. This advantage is more significant when graphs are more complicated since the structure contains more information. Neglecting this information would make the SC-DGML method possess large-scale of prediction errors.

In the second part, the assumption of the known structure is preserved, and the performances of these two methods in terms of different proportions of missing data and in various data-missing modes are studied. Specifically, the proportion of missing data is set to four levels: $p_{miss} = \{0, 0.25,\ 0.5,\ 0.75\}$ and three modes of missing data are considered: (a) random missing, which means we randomly choose $\mathcal{T}_{miss}$ and $|\mathcal{T}_{miss}|$ to be the same for each signal; (b) imbalanced missing, which means we randomly delete p_{miss} proportions of data from the matrix instead of signals but keep $|\mathcal{T}_{miss}| \in (0 \times |\mathcal{T}|, 0.75 \times |\mathcal{T}|)$ for each signal; and (c) non-uniform missing, which means we randomly delete p_{miss} proportions of continuous data from each signal. For simplicity, studies of different missing settings are conducted in G1 with $\sigma = 0.1$, and detail results are shown in Table 3.

This table shows that modes of missing data have little influence on the results, which assures the stability of the proposed method in various situations. Additionally, compared with SC-DGML, the proposed method is more robust when a large proportion of data is missing. When little data are missing, the prediction errors of the two methods are close since the observed data are sufficient to construct a precise model and the advantage of GBSC in terms of signal completion is less significant.

In the third part, without the known-structure assumption, the performances of the methods in terms of structure learning are explored. Apart from the prediction accuracy, the similarity between the learned graph and the true graph is also considered. Comparisons of the learned graphs and original graphs of G1 and G2 are shown in Figure 6, with the relative confusion matrices listed in Tables 4 and 5. The proportion of missing data is set to $p_{miss} = 0.5$, and the scale of error is $\sigma = 0.1$.

The learned graphs from the SC-DGML method of G1 and G2 when $p_{miss} = 0.5$ are both fully connected graphs, that is, $\hat{\mathcal{A}}_{SC-DGML} = \{(i,j)|i,j \in \mathcal{N}, i < j\}$, and therefore the results are not

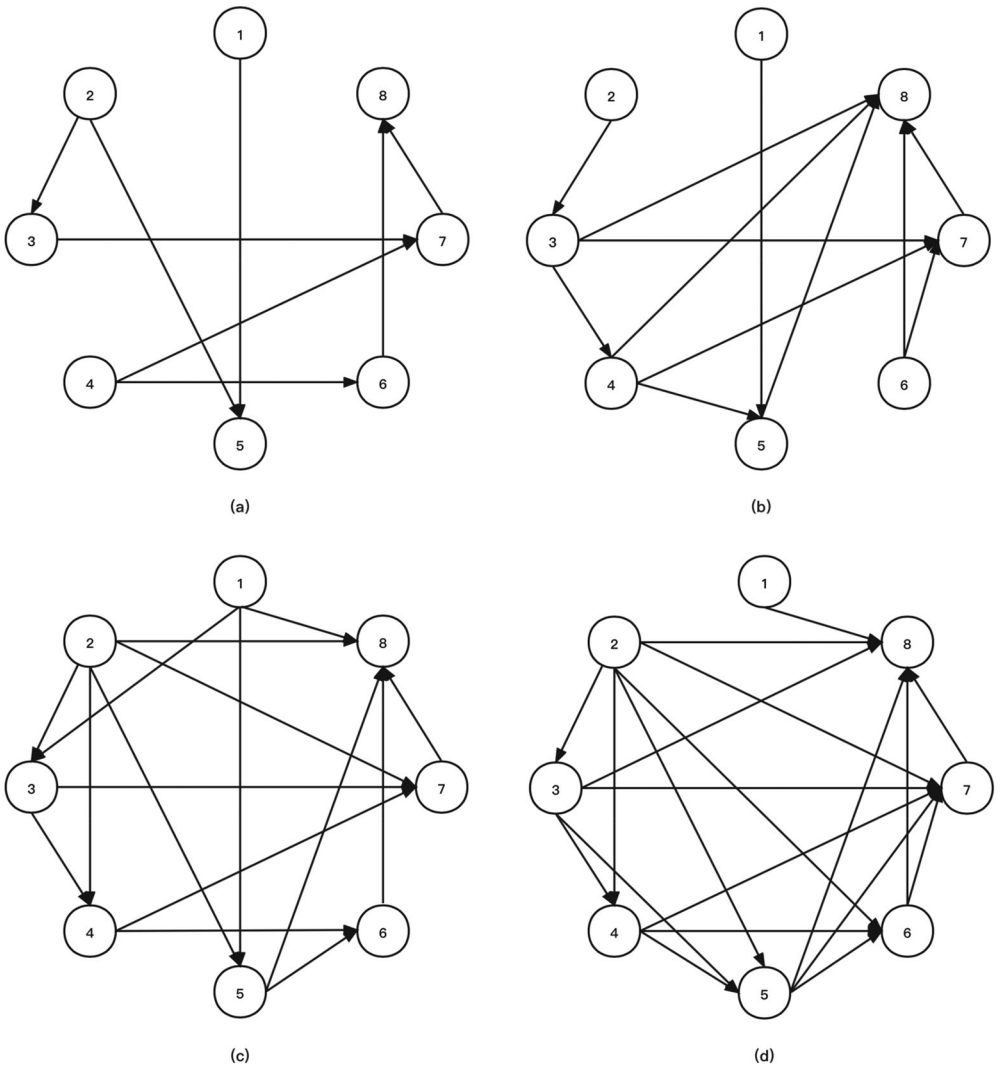

Figure 6. Comparisons of the graphs learned by the GBSC method and the original graphs when $p_{miss} = 0.5$. (a) Original G1; (b) learned G1; (c) original G2; and (d) learned G2.

Table 4. Confusion matrix of the learned G1 for the GBSC method ($p_{miss} = 0.5$).

		Predicted graph	
G1: 8 nodes, 8 arcs		True	False
True graph	True	6	2
	False	6	14
F_1 score		0.6	

Table 5. Confusion matrix of the learned G2 for the GBSC method ($p_{mis} = 0.5$).

		Predicted graph	
G2: 8 nodes, 16 arcs		True	False
True graph	True	14	2
	False	6	6
F_1 score		0.778	

Table 6. Prediction accuracy of the two methods with different missing proportions of data in G1 and G2 when the structure is unknown (standard errors are shown in parenthesis).

			Missing proportions			
	Methods	Scale of error	0	0.25	0.5	0.75
G1	GBSC	0.1	0.529(0.03)	0.528(0.05)	0.673(0.04)	0.738(0.16)
		0.5	0.715(0.37)	0.743(0.56)	0.816(0.41)	0.921(0.25)
	SC-DGML	0.1	0.526(0.04)	0.530(0.03)	0.678(0.05)	3.098(6.88)
		0.5	0.711(0.28)	0.906(1.29)	1.341(1.57)	16.706(33.41)
G2	GBSC	0.1	0.390(0.22)	0.394(0.07)	0.716(0.27)	1.594(4.36)
		0.5	0.603(0.48)	0.728(0.71)	1.190(1.09)	1.281(0.27)
	SC-DGML	0.1	0.370(0.18)	0.409(0.07)	0.726(0.33)	4.506(17.15)
		0.5	0.623(0.55)	0.951(1.03)	2.607(4.57)	24.175(49.68)

shown in detail. As shown in a previous work (Estrada, Paynabar, and Pacella 2021), the learned graphs of the SC-DGML method tend to add more arcs that do not exist in the original graphs, and our results show that this effect is greater when the signal matrices are incomplete. A reasonable explanation is that in the SC-DGML method, the matrices are completed without information from potential F-to-F regression relationships, which would reduce correlations from variables between nodes and make it difficult for group LASSO to select variables that are truly related. In the proposed GBSC method, learned graphs are sparser compared with the SC-DGML method, resulting from the adjusted MMMF and recursive improvements. Although the learned graphs might miss some of the existing arcs, this relatively sparser graph is more useful in real-world applications when the graph structure of the system is unavailable. Additionally, the prediction accuracy performance under different circumstances when the graph structure is unknown is shown in Table 6.

Most of the findings are similar to the case when the structure is known, except that the methods have better prediction accuracy when the arcs are denser. Consider a simple graph with $G = \{\mathcal{N} = \{1,2,3\},\ \mathcal{A} = \{(1,2),\ (2,3)\}\}$. When the structure is known, the data of node 3 are predicted only by the data from node 2. However, when the structure is unknown, the arc $(1,3)$ might be learned because of the transmission effect. The added arc might actually help better predict the data of node 3, and this effect occurs more often in denser graphs. Nevertheless, the proposed GBSC method is more stable and accurate than SC-DGML in structure learning, especially when a large proportion of data is missing.

Finally, we present a further display of how the proposed method recurrently obtains a more accurate prediction as more information about the graph structure becomes available. When learning the graph structure, the completed signal by the GBSC method becomes more similar to the original signal as the number of iterations increases. Additionally, the learned graph also

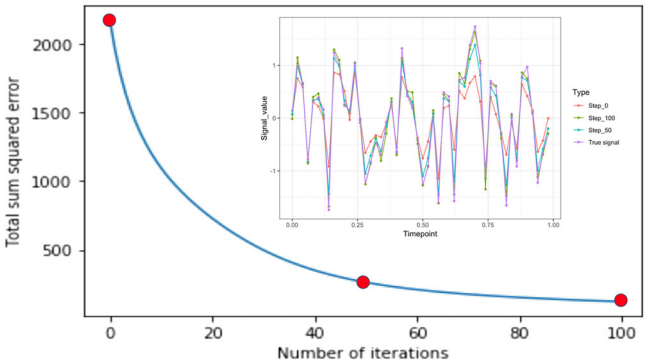

Figure 7. Decreasing curve of error and comparisons of the completed signal and original signal in the GBSC method (average of all replications in node 5 of G1 and $p_{miss} = 0.5$).

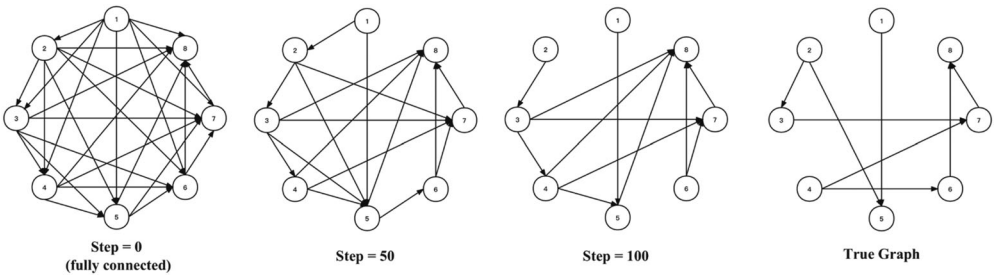

Figure 8. Evolution of the learned graph.

becomes denser and closer to the true graph. Figure 7 shows the decreasing curve of errors by the GBSC method between the completed matrix and the original matrix in node 5 of G1 with missing proportions $p_{miss} = 0.5$ and $\sigma = 0.1$, together with comparisons between the completed signal and the original signal at different iteration numbers $Step = \{0, 50, 100\}$. The evolution of the learned graph at these iteration points of the whole G1 graph is shown in Figure 8. Note that the result when $Step = 0$ is the same as that from the SC-DGML method.

This decreasing curve of errors can also help determine the parameter of the maximum number of iterations to avoid redundant computational costs in real-world applications. When complete matrices are unavailable, changes of two continuous estimation matrices could be calculated as follows, which is similar with differentials of the decreasing curve.

$$e^{(t)} = \|\boldsymbol{S}_m^{(t)} - \boldsymbol{S}_m^{(t-1)}\|_F^2 \tag{31}$$

Then the curve of $e^{(t)}$ can be drawn, and the maximum number of iterations can be chosen so that $e^{(t_{\max})}$ is close enough to 0 or below an acceptable threshold.

4. Real-world example

In this section, the proposed GBSC method is applied to the manufacturing process of monocrystalline silicone ingots. As mentioned in Sec. 1, the CZ method has become the mainstream method of monocrystalline silicone production due to the advantages of high-level equipment automation, simplicity, and efficiency. On the other hand, the CZ method has a relatively longer cycle time, larger material cost, and higher quality requirement. Typically, there are many sensors placed in the system that continuously collect data regarding the variables used for system monitoring and control. In addition, the CZ method consists of several stages, during which the

relationship among the variables might differ. Therefore, to reduce the heavy loss incurred by failure events, it is essential to learn the variable relationship structures in the harsh real-world environments. The two most important and complicated stages, melting and shoulder growth, are studied in this section.

As the two stages have different physical and chemical mechanisms, the variables that can be used to characterize the two stages and their relationship also change. In this study, the raw data contain more than 50 variables. We first split the data records by stages and then remove variables that have almost constant readings throughout each stage. Finally, nine variables from the melting stage, denoted as M1–M9, and 13 from the shoulder growth, denoted as S1–S13, are retained. These include important variables that can characterize the system status and production progress, such as diameter measurements, thermal field temperatures, main heater currents and throttle valve outputs, etc. Signals from several nodes of the two stages are shown in Figures 9 and 10.

The signal data are collected from 70 ingot samples. We divide the samples as follows: $N_{train} = 40$, $N_{valid} = 10$, and $N_{test} = 20$. The lengths of signals vary from samples, $L_{mi} \in (90, 200)$, and therefore, all signals are preprocessed such that $L_{mi} = L_m$. Since there are multiple samples with different lengths, traditional signal alignment methods like dynamic time warping (DTW) cannot be easily used. In this case, a straightforward alignment method is used that all signal samples are first scaling into one unit of time, for example, signal from ith sample with L_{mi} grid points are processed into $\tilde{\boldsymbol{x}}_{mi} = \left[\mathrm{x_{mi}}(0), x_{mi}\left(\frac{1}{L_{mi}-1}\right), \ldots, x_{mi}\left(\frac{L_{mi}-2}{L_{mi}-1}\right), x_{mi}(1)\right]$. After a piecewise-linear function $f_{mi}(t) : [0,1] \to \mathbb{R}$ is estimated by these L_{mi} points, we can align all sample signals by taking L_m points as $\boldsymbol{x}_{mi} = \left[f_{\mathrm{mi}}(0), f_{mi}\left(\frac{1}{L_m-1}\right), \ldots, f_{mi}\left(\frac{L_m-2}{L_m-1}\right), f_{mi}(1)\right]$. In the monocrystalline growth process, the length difference mainly comes from different amount of raw materials, that is, polycrystalline silicon. Also, signals from different samples have similar shape and trend, with few significant or high-frequency fluctuation as shown in Figures 9 and 10, which makes the preprocessing method sensible for little information loss. The number of extracted PCs A_m is set according to the

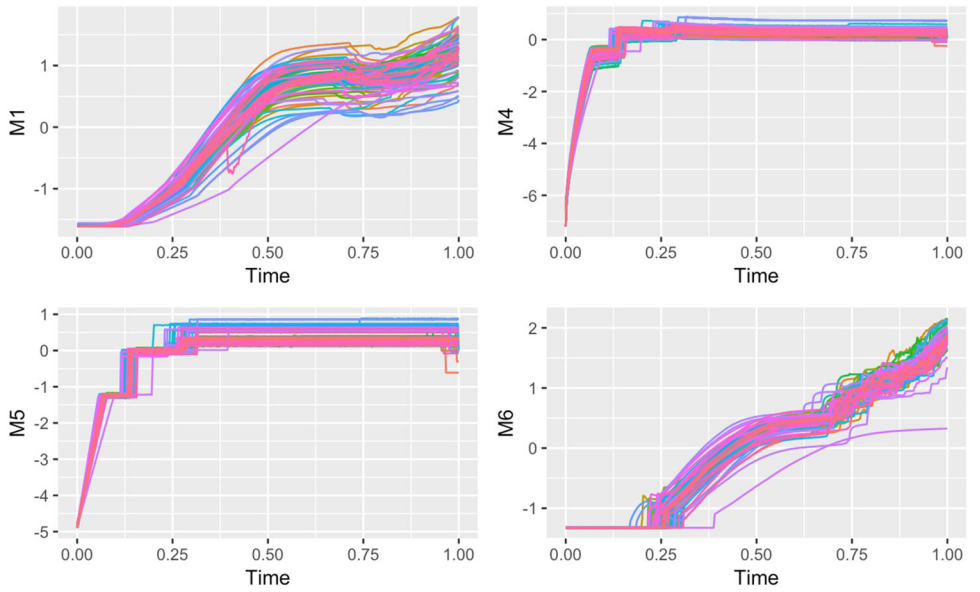

Figure 9. Signals for some variables involved in the melting stage (each color represents the signal from each monocrystalline silicone ingot).

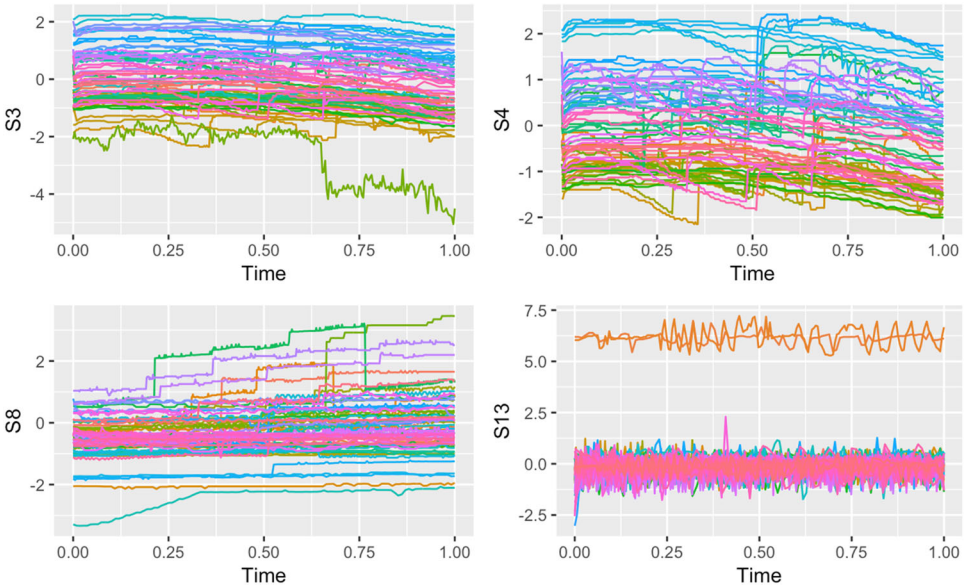

Figure 10. Signals for some variables involved in the shoulder growth stage (each color represents the signal from each monocrystalline silicone ingot).

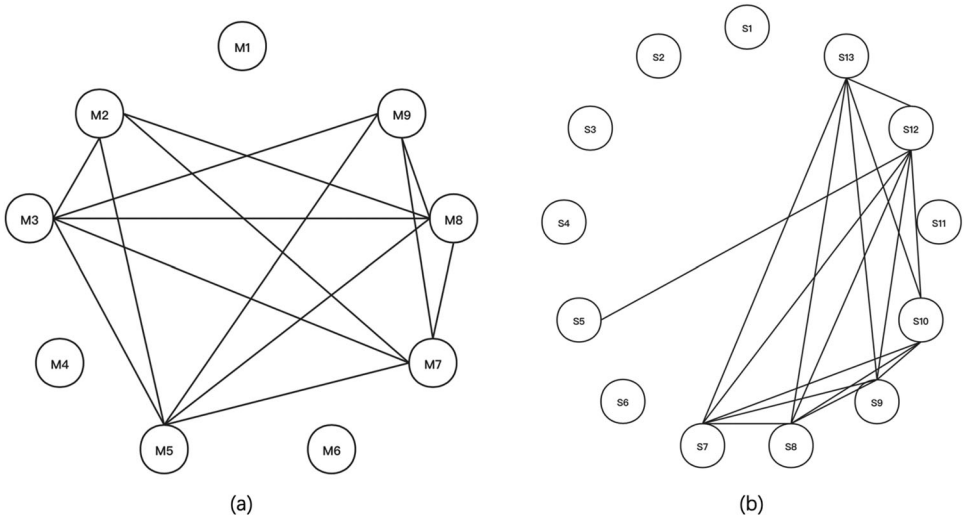

Figure 11. Graphs learned by the GBSC method when $p_{miss} = 0.5$. (a) Shoulder growth stage; and (b) melting stage.

criterion of PCA using complete data from different nodes, and the dimensions of low-rank matrices, K_m, are set equal to A_m. The candidate sets of the penalty parameters are $C_\lambda = \{10^{-12}, 10^{-10}, 10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$ and $C_\mu = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$. After randomly deleting the $p_{miss} = 0.5$ proportion of data for each signal, the learned graphs of the proposed GBSC method are shown in Figure 11.

In this example, the arcs (directed) are replaced by edges (undirected) in the two learned graphs since no domain knowledge regarding the directions of possible relationships between variables is available. The learned graph in this case is set to be undirected because the aim of using DGM learning in this case is to learn the potential relationship between variables instead of causal relationship or control relationship in the process. Since the actual process relationship

might contain loops or both-way feedbacks, which might violate the DGM assumption. To avoid the learned graphs containing loops, the constraints that $l < m$ is added, and this idea comes from the previous work when learning DGMs (Estrada, Paynabar, and Pacella 2021). Therefore, the candidate parents of each node are set as: $Cpa(m) = \{l \in \mathcal{N}, l < m\}$, and the learned edges only suggest that there is a possible relationship between these two variables. Similar to the results from the numerical simulations, the graphs learned by the GBSC method are relatively sparser and therefore provide meaningful guidance in terms of the potential structure of the real system. Note that the learned graph clearly separates all variables into two groups. The first group consists of isolated variables that have no edge connections with any other nodes. For example, in the melting stage, the thermal field temperature, main heater current (M4), and liquid level temperature (M6) are relatively stable and are not affected by (or affect) any other variables; in the shoulder growth stage, the thermal field temperature (S3), and liquid level temperature (S6) are isolated from the others. The second group consists of variables that are pairwise connected to each other. For example, in the melting stage, the main room pressure (M2) and minor room pressure (M3), as well as the main heater power measurement (M5) and thermal field setpoint (M9) are connected; in the shoulder growth stage, the main room pressure (S13) and throttle valve output (S10) are connected. Further verification with engineering knowledge confirms that these identified connections are meaningful in practice.

In addition, some variables, such as the thermal temperature (denoted as M1 and S2 in two stages) and main heater current (M4 and S3), remain isolated in both stages. This observation shows that the algorithm is stable in performance since the two graphs are learned separately using different datasets. Similarly, the main room pressure (M2 and S13), throttle valve opening measurement (M7 and S8), and throttle valve output (M8 and S10) are mutually connected to each other in both stages. These findings are also consistent with expert knowledge.

It is also interesting that the variable relationship changes across the two stages. For example, minor room pressure (M3) was in the connect group in the melting stage but was not in the graph of the growth stage. This change could be explained by the fact that the variable is critical and adjusted frequently during the melting stage but remains almost constant during shoulder growth and was excluded from analysis during data preprocessing. In general, these data-driven findings regarding the relationships among variables are helpful to engineers when constructing the true structure of the manufacturing system.

5. Conclusions

In this study, a graph-based signal completion method is used to learn the functional DGM with incomplete signals, including the functional relationship between nodes and the graph structure. The proposed method is an extension of the MMMF method, which integrates information from the F-to-F regression and graph structure, and it adds a penalty term that maintains the regression effect when filling incomplete matrices. Specifically, compared to other matrix completion methods that only consider observed data and low-rank assumptions, the added term possesses potential structural information and can be updated as the completion becomes more accurate. The results of the numerical experiments and a real-world case study if monocrystalline silicone manufacturing verify the efficiency of the proposed method in both scenarios, that is, in the case of a known structure and unknown structure, compared with the SC-DGML method. This enhanced performance is especially true when the missing proportion of data is relatively large. In terms of future research, theoretical proof of convergence of the recursive framework needs to be studied. Additionally, extension of the method to learn nonlinear functional relationships is needed by many real complex.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was funded by the Key Program of National Science Foundation of China under Grant No. 71932006.

Data availability statement

Due to the nature of this research, participants of this study did not agree for their data to be shared publicly, so supporting data is not available.

References

Balzano, L., R. Nowak, and B. Recht. 2010. Online identification and tracking of subspaces from highly incomplete information. In 2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton), Allerton House, Monticello, Illinois, September 29-October 1. doi:10.1109/ALLERTON.2010.5706976.

Cao, X., B. Sandstede, and X. Luo. 2019. A functional data method for causal dynamic network modeling of task-related fMRI. *Frontiers in Neuroscience* 13:127. doi:10.3389/fnins.2019.00127.

Chen, T., W. Zhang, Q. Lu, K. Chen, Z. Zheng, and Y. Yu. 2012. SVDFeature: A toolkit for feature-based collaborative filtering. *The Journal of Machine Learning Research* 13:3619–22.

Estrada, A., K. Paynabar, and M. Pacella. 2021. Functional directed graphical models and applications in root-cause analysis and diagnosis. *Journal of Quality Technology* 53 (4):421–37. doi:10.1080/00224065.2020.1805380.

Fang, X., H. Yan, N. Gebraeel, and K. Paynabar. 2021. Multi-sensor prognostics modeling for applications with highly incomplete signals. *IISE Transactions* 53 (5):597–613. doi:10.1080/24725854.2020.1789779.

Guo, X., and H. Zhang. 2021. Structured learning of time-varying networks with application to PM2.5 data. *Communications in Statistics - Simulation and Computation* 50 (5):1364–82. doi:10.1080/03610918.2019.1582780.

Han, S. W., S. Park, H. Zhong, E.-S. Ryu, P. Wang, S. Jung, J. Lim, J. Yoon, and S. Kim. 2021. Estimation of joint directed acyclic graphs with lasso family for gene networks. *Communications in Statistics - Simulation and Computation* 50 (9):2793–807. doi:10.1080/03610918.2019.1618869.

Hastie, T., R. Tibshirani, and M. Wainwright. 2015. *Statistical learning with sparsity: The Lasso and generalizations*, 367. Boca Raton, Florida: Chapman & Hall/CRC.

Horváth, L., and L. Horváth. 2012. *Inference for functional data with applications*, vol. 200. Berlin, Germany: Springer.

Langseth, H., and L. Portinale. 2007. Bayesian networks in reliability. *Reliability Engineering & System Safety* 92 (1):92–108. doi:10.1016/j.ress.2005.11.037.

Li, B., and E. Solea. 2018. A nonparametric graphical model for functional data with application to brain networks based on fMRI. *Journal of the American Statistical Association* 113 (524):1637–55. doi:10.1080/01621459.2017.1356726.

Matsuda, T., and F. Komaki. 2019. Empirical Bayes matrix completion. *Computational Statistics & Data Analysis* 137:195–210. doi:10.1016/j.csda.2019.02.006.

Pourhabib, A., J. Z. Huang, and Y. Ding. 2016. Short-term wind speed forecast using measurements from multiple turbines in a wind farm. *Technometrics* 58 (1):138–47. doi:10.1080/00401706.2014.988291.

Qiao, X., S. Guo, and G. M. James. 2019. Functional graphical models. *Journal of the American Statistical Association* 114 (525):211–22. doi:10.1080/01621459.2017.1390466.

Sun, H., S. Huang, and R. Jin. 2017. Functional graphical models for manufacturing process modeling. *IEEE Transactions on Automation Science and Engineering* 1:1–10.

Sun, J., H. Liao, and B. R. Upadhyaya. 2014. A robust functional-data-analysis method for data recovery in multi-channel sensor systems. *IEEE Transactions on Cybernetics* 44 (8):1420–31. doi:10.1109/TCYB.2013.2285876.

Takács, G., and D. Tikk. 2012. Alternating least squares for personalized ranking. In Proceedings of the sixth ACM conference on Recommender systems. Association for Computing Machinery: Dublin, Ireland. p. 83–90. doi:10.1145/2365952.2365972.

Tan, W. J., A. N. Zhang, and W. Cai. 2019. A graph-based model to measure structural redundancy for supply chain resilience. *International Journal of Production Research* 57 (20):6385–404. doi:10.1080/00207543.2019.1566666.

Toh, K.-C., and S. Yun. 2010. An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. *Pacific Journal of Optimization* 6:178–90.

Wang, H. 2021. Zipf matrix factorization: Matrix factorization with Matthew effect reduction. In 2021 4th International Conference on Artificial Intelligence and Big Data (ICAIBD), Chengdu, China, 28-31 May. doi:10.1109/ICAIBD51990.2021.9459018.

Xia, L., P. Zheng, X. Li, R. X. Gao, and L. Wang. 2022. Toward cognitive predictive maintenance: A survey of graph-based approaches. *Journal of Manufacturing Systems* 64:107–20. doi:10.1016/j.jmsy.2022.06.002.

Yan, H., J. Wang, J. Chen, Z. Liu, and Y. Feng. 2022. Virtual sensor-based imputed graph attention network for anomaly detection of equipment with incomplete data. *Journal of Manufacturing Systems* 63:52–63. doi:10.1016/j.jmsy.2022.03.001.

Yuan, M., and Y. Lin. 2007. Model selection and estimation in the Gaussian graphical model. *Biometrika* 94 (1): 19–35. doi:10.1093/biomet/asm018.

Zhou, R., N. Gebraeel, and N. Serban. 2012. Degradation modeling and monitoring of truncated degradation signals. *IIE Transactions* 44 (9):793–803. doi:10.1080/0740817X.2011.618175.

Zhu, H., N. Strawn, and D. B. Dunson. 2016. Bayesian graphical models for multivariate functional data. *Journal of Machine Learning Research* 17 (1):7157–83.

Zou, H., and T. Hastie. 2005. Regularization and variable selection via the elastic nets. *Journal of the Royal Statistical Society Series B: Statistical Methodology* 67 (2):301–20. doi:10.1111/j.1467-9868.2005.00503.x.