Multivariate Bernoulli Logit-Normal Model for Failure Prediction

Huijuan Shao[1], Xinwei Deng[2], Chi Zhang[3], Shuai Zheng[4], Hamed Khorasgani[5], Ahmed Farahat[6], and Chetan Gupta[7]

[1,3,4,5,6,7] *Industrial AI Lab, Hitachi America, Ltd. R&D, Santa Clara, CA, 95054, U.S.A.*
huijuan.shao@hal.hitachi.com, chi.zhang@hal.hitachi.com, shuai.zheng@hal.hitachi.com,
hamed.khorasgani@hal.hitachi.com, ahmed.farahat@hal.hitachi.com, chetan.gupta@hal.hitachi.com

[2] *Department of Statistics, Virginia Tech, Blacksburg, VA, 24060, U.S.A.*
xdeng@vt.edu

ABSTRACT

The failures among connected devices that are geographically close may have correlations and even propagate from one to another. However, there is little research to model this problem due to the lacking of insights of the correlations in such devices. Most existing methods build one model for one device independently so that they are not capable of capturing the underlying correlations, which can be important information to leverage for failure prediction. To address this problem, we propose a multivariate Bernoulli Logit-Normal model (MBLN) to explicitly model the correlations of devices and predict failure probabilities of multiple devices simultaneously. The proposed method is applied to a water tank data set where tanks are connected in a local area. The results indicate that our proposed method outperforms baseline approaches in terms of the prediction performance such as receiver operating characteristic curve.

1. INTRODUCTION

Failure prediction is an important problem in industry and has been studied over decades in various areas. Generally, majority of equipments, and industrial components deteriorate after running for a period of time. The failures among multiple devices, which are physically connected to each other, may propagate. When a component of a system fails, other relevant components may break down too. For example in a mill plant, when a motor fails, the bearings, which are physically connected to it, may fail as well. Although the properties of these multiple devices are different, they may have correlations to each other. However, there is little research to provide approaches to solve this critical issue. In this work, we intend to study this problem by modeling the relationship between devices or components.

Huijuan Shao et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 United States License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

In practice, multiple devices installed in the same location may have failures at the same time. In many cases, the relationship between the devices' measurements may have big impact in predicting their failures. To capture these relationships, model-based techniques use system equations to extract analytical redundancies between devices' measurements (Ragot & Maquin, 2006),(Ferrari, Parisini, & Polycarpou, 2011). Unfortunately, for complex systems, the system models are not easy to develop. Moreover, the environment keeps on changing during the system life-cycle. Therefore, reliable models are not always available (Khorasgani, 2017). An alternative approach is to apply a data-driven solution that uses information from similar devices for failure prediction. Data-driven methods use system's measurements as the features for failure prediction (Khorasgani, Farahat, Ristovski, Gupta, & Biswas, 2018), (Zheng, Ristovski, Farahat, & Gupta, 2017), (Q. Wang, Zheng, Farahat, Serita, & Gupta, 2019). In order to predict multivariate responses from multiple devices simultaneously with higher accuracy, we use the correlated features from these devices. The common features are either explicitly or implicitly hidden in the data, which is captured by sensors or monitored events. Compared to only utilizing the data of each device or component, the combined information from multiple devices or components can borrow strength from each other.

How to use such common features is a challenge. To the best of our knowledge, there is a lack of adequate methods to leverage correlation for model building. The majority of research builds an individual model for each type of devices and predicts the failure for each device separately. A common research direction is multi-task learning (Gong, Ye, & Zhang, 2012a), (Gong, Ye, & Zhang, 2012b), (Gong, Zhou, Fan, & Ye, 2014), (Jalali, Sanghavi, Ruan, & Ravikumar, 2010). These methods build a model for each type of devices or components then aggregate these models and extract similarity among models. Another research direction is to utilize regularization to estimate the $p \times q$ regression coefficients and

obtain the correlation among predictors (Liu, Wang, & Zhao, 2014) , (Lozano, Jiang, & Deng, 2013), (Rothman, Levina, & Zhu, 2010), (W. Wang, Liang, & Xing, 2013). But the responses in all these papers are continuous multivariate variables rather than multivariate binary variables, which are not applicable to failure prediction. Other researchers employ existing network structure of physical models (Khorasgani, Farahat, Hasanzade, & Gupta, 2019) as one of the inputs for failure prediction. However, in reality we may not be able to obtain the graph structure inside a system or among devices.

In this paper, we introduce multi-Bernoulli distribution with logit transformation to learn the correlation between the predictors and multivariate responses. We handle the dependency of multivariate response through the analysis between one device and another device. Therefore, we build a single model for multiple devices at once rather than creating multiple models. The advantages of our model over other models are listed as below. First, it can capture the correlation among different physical devices by computing a coefficient matrix and an inverse covariance matrix. The learned correlation can help to interpret the underlying correlation among multiple devices, even with significantly different physical properties. Thus, it may help us have more insight into the domain knowledge. Second, our method generates a unified model rather than a combination of multiple models. This makes model development process much simpler. Having a single model for several devices, can simplify model management task and reduce cloud computing costs during the application. Moreover, our unified model can be applied to multiple types of devices.

This paper is organized as follows: Section 2 briefly describes the multivariate Poisson log-normal model (MVPLN), on which this model is proposed. In Section 3, we formulate the concurrent multiple devices failure prediction problem as a multivariate Bernoulli logit-normal model. Section 4 focuses on estimating the parameters of this model using the Monte Carlo expectation maximization algorithm. In Section 5, we apply this MBLN model to a water tank dataset generated by a simulator. Section 6 discusses the pros and cons of this approach, and proposes future work.

2. BACKGROUND

There are a few approaches which can capture the correlation of devices with different features. A comparable approach is MVPLN (Wu, Deng, & Ramakrishnan, 2018). MVPLN model was proposed to solve the prediction problem where both predictors and responses are count data. In order to link the input variables $\{x_1, \ldots, x_p\}$ to the output discrete variables $\{y_1, \ldots, y_q\}$, it builds a linear regression model and introduces a latent variable to reflect this relationship. Since the responses are count data, the MVPLN model assumes that the responses conform to multivariate Poisson distribution. The objective function of MVPLN is a sum of expected joint likelihood function and l_1 penalties on two model parameters. To reach the optimal value of the objective function, it utilizes a Monte Carlo expectation maximization (EM) algorithm (Moon, 1996) to estimate the model parameters. In the E-step, an expected log-likelihood function is formulated by Monte Carlo techniques. In the M-step, the optimization problem is not convex. Thus, it uses an iterative and alternative approach by fixing a model parameter and solving another one. In each iteration, a model parameter is solved by Lasso (Tibshirani, 1996) and another one is tackled by Graph Lasso (Friedman, Hastie, & Tibshirani, 2008). This research is motivated by multiple devices' failure prediction, where the responses are multivariate binary variables. We assume the responses conform to certain distribution multivariate Bernoulli distribution as multivariate Poisson distribution in MVPLN model. Different from the log link function in MVPLN, we use the logit function as link function instead in order to predict binary variables.

This MBLN model overcomes two drawbacks: over-dispersion and zero-inflation. For over-dispersion, the responses in that paper spread over the whole integer space. With Poisson distribution to simulate the response, the variance would be very large. However, Poisson distribution only has one free parameter. Researchers cannot adjust the variance independent of the mean, i.e. Poisson distribution over-dispersion problem. In the MVPLN model, zero-inflation appears in the sampling step of the Metropolis-hasting algorithm, during which a lot of negative values are sampled but discarded. MBLN avoids the zero-inflation problem because of two reasons. (1) The response of this model falls between 0 and 1. The variance of the response is small. (2) This model samples from the multivariate-normal distribution directly rather than using the metropolis-hasting algorithm, which eliminates the need to discard useless samples.

3. PROBLEM DEFINITION AND FORMULATION

The follow notations are used in this paper. Lower case letters, such as x and y denote scalers, whereas bold lower case letters such as $\boldsymbol{x}$ and $\boldsymbol{y}$ represent vectors. The j^{th} component of the vector $\boldsymbol{x}$ is shown as a lower case letter with subscript x_j. Bold calligraphic upper case letters $\mathcal{X}$ and $\mathcal{Y}$ denote random column vectors. Bold norm upper case letters $\boldsymbol{X}$ and $\boldsymbol{Y}$ stand for the matrices. The (j, k) entry of matrix $\boldsymbol{Y}$ is expressed as $y_{j,k}$.

Multivariate Bernoulli Logit-Normal Model

The input are features from multiple devices. These features can be discrete or continuous variables. The output are multivariate binary response for different devices. This output can be represented as a multivariate random variable $\mathcal{Y} = [\mathcal{Y}_1, \mathcal{Y}_2, ..., \mathcal{Y}_q]^T \in \mathcal{Z}_{+q}$, where the superscript T denotes the transpose, and $\mathcal{Z}_+$ denotes the set of binary integer

variables. We assume that the binary response $\mathcal{Y}$ follows the multivariate Bernoulli distribution. Each dimension of $\mathcal{Y}$, i.e. $\mathcal{Y}_k$, follows the univariate Bernoulli distribution with parameter θ_k. Thus, any dimension $\mathcal{Y}_k$ is conditionally independent of other dimensions given θ_k.

$$\begin{aligned} \mathcal{Y}_k \sim Bern(\theta_k) = (\theta_k)^{y_k}(1-\theta_k)^{1-y_k}. \\ \theta_k \in (0,1), \forall k = 1,2,...,q \end{aligned} \tag{1}$$

Given the predictor vector $\boldsymbol{x} = \{x_1, x_2, ..., x_p\}^T \in R_p$, we use a regression model to connect the relationship between $\mathcal{Y}$ and $\boldsymbol{x}$ as Equation 2.

$$\boldsymbol{\gamma} = \begin{bmatrix} \log\frac{\theta_1}{1-\theta_1} \\ \vdots \\ \log\frac{\theta_k}{1-\theta_k} \\ \vdots \\ \log\frac{\theta_q}{1-\theta_q} \end{bmatrix} = \boldsymbol{B}^T\boldsymbol{x} + \epsilon, \epsilon \sim N(0, \boldsymbol{\Sigma}),$$

where $\boldsymbol{B}$ is a $p \times q$ coefficient matrix, and $\boldsymbol{\Sigma}$ denotes a $q \times q$ covariance matrix which represents the covariance structure of variable $\boldsymbol{\theta}$. With the conditionally independence assumption, the probability mass function for the multivariate Bernoulli random variable $\boldsymbol{y}$ is

$$p(\boldsymbol{\mathcal{Y}} = \boldsymbol{y}|\boldsymbol{\theta}) = \prod_{k=1}^{q} p(\mathcal{Y}_k = y_k|\theta_k) = \prod_{k=1}^{q} (\theta_k)^{y_k}(1-\theta_k)^{1-y_k}. \tag{2}$$

Here $\boldsymbol{\theta} = [\theta_1, \theta_2, ..., \theta_q]^T$ given $\boldsymbol{x}$. $\boldsymbol{\gamma} = \log\frac{\boldsymbol{\theta}}{1-\boldsymbol{\theta}}$ follows the multivariate Gaussian distribution $N(\boldsymbol{B}^T\boldsymbol{x}, \boldsymbol{\Sigma})$ with density function

$$p(\boldsymbol{\gamma}|\boldsymbol{x}) = \frac{1}{(2\pi)^{q/2}|\boldsymbol{\Sigma}|^{1/2}} exp(-\frac{1}{2}(\boldsymbol{\gamma} - \boldsymbol{B}^T\boldsymbol{x})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\gamma} - \boldsymbol{B}^T\boldsymbol{x})). \tag{3}$$

Therefore, we derive the density function of $\boldsymbol{\theta}|x$ as Equation 4.

$$\begin{aligned} p(\boldsymbol{\theta}|x) &= p_{\boldsymbol{\gamma}}(\log(\frac{\boldsymbol{\theta}}{1-\boldsymbol{\theta}})|x)|diag(\frac{1}{\theta_k(1-\theta_k)})| \\ &= \frac{exp(-\frac{1}{2}(\log\frac{\boldsymbol{\theta}}{1-\boldsymbol{\theta}} - \boldsymbol{B}^T\boldsymbol{x})^T\boldsymbol{\Sigma}^{-1}(\log\frac{\boldsymbol{\theta}}{1-\boldsymbol{\theta}} - \boldsymbol{B}^T\boldsymbol{x}))}{(2\pi)^{q/2}|\boldsymbol{\Sigma}|^{1/2}\prod_{k=1}^{q}\theta_k(1-\theta_k)}. \end{aligned} \tag{4}$$

With n number of the predictors $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n]^T$ and responses $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_n]^T$, the log-likelihood of the multivariate Bernoulli logit-normal model is computed as Equation 5.

$$L(\boldsymbol{B}, \boldsymbol{\Sigma}) = \sum_{j=1}^{n} \log p(\boldsymbol{\mathcal{Y}} = \boldsymbol{y}_j|\boldsymbol{x}_j), \tag{5}$$

where,

$$p(\boldsymbol{\mathcal{Y}} = \boldsymbol{y}|\boldsymbol{x}) = \int_{\boldsymbol{\theta}} p(\boldsymbol{\mathcal{Y}} = \boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{x})d\boldsymbol{\theta}. \tag{6}$$

Here, $p(\boldsymbol{\mathcal{Y}} = \boldsymbol{y}|\boldsymbol{\theta})$ and $p(\boldsymbol{\theta}|\boldsymbol{x})$ follow the multivariate Bernoulli distribution and multivariate logit-normal distribution. In order to derive the coefficient matrix $\boldsymbol{B}$ and the inverse covariance matrix $\boldsymbol{\Sigma}^{-1}$, we introduce l_1 to penalize these two model parameters. Therefore the loss function becomes as Equation 7.

$$L_p(\boldsymbol{B}, \boldsymbol{\Sigma}) = -L(\boldsymbol{B}, \boldsymbol{\Sigma}) + \lambda_1||\boldsymbol{B}||_1 + \lambda_2||\boldsymbol{\Sigma}^{-1}||_1, \tag{7}$$

where $\|\cdot\|_1$ denote the l_1 matrix norm, which is defined as $\|\boldsymbol{B}\|_1 = \boldsymbol{\Sigma}_{j,k}|b_{j,k}|$, and $\lambda_1 > 0$, $\lambda_2 > 0$ are two tuning parameters.

4. MONTE CARLO EM ALGORITHM FOR PARAMETER ESTIMATION

The paper (Wu et al., 2018) uses a Monte Carlo expectation maximization (MCEM) algorithm to estimate the model parameters $\boldsymbol{B}$ and $\boldsymbol{\Sigma}$. This MBLN model utilizes a similar MCEM algorithm to approximate a numerical solution for the same two model parameters. Also, we adopt the same criteria EBIC to select the turning parameters λ_1, λ_2. In the E-step, MBLN uses the logit function as link function rather than log function in that paper. Therefore, the formulation and derivation are different when deriving the log-likelihood function.

4.1. Monte Carlo (MC) E-step

In the iteration $t+1$ of the MC E-step, in order to obtain the conditional probability distribution of $\boldsymbol{\theta}_j = [\theta_{j1}, ..., \theta_{jk}, ..., \theta_{jq}]^T$, we use a $m \times q$ matrix $\boldsymbol{\Theta}_j = [\boldsymbol{\theta}_j^{(1)}, ..., \boldsymbol{\theta}_j^{(\tau)}, ..., \boldsymbol{\theta}_j^{(m)}]^T$ from $p(\boldsymbol{\theta}_j|\boldsymbol{\mathcal{Y}} = \boldsymbol{y}_j, \boldsymbol{x}_j; \boldsymbol{B}^{(t)}, \boldsymbol{\Sigma}^{(t)})$ to estimate the expected log-likelihood function.

$$\begin{aligned} \tilde{Q}(\boldsymbol{B}, \boldsymbol{\Sigma}|\boldsymbol{B}^{(t)}, \boldsymbol{\Sigma}^{(t)}) = \sum_{j=1}^{n}\frac{1}{m}\sum_{\tau=1}^{m}[\log p(\boldsymbol{\mathcal{Y}} = \boldsymbol{y}_j, \boldsymbol{\theta}_j^{(\tau)}|\boldsymbol{x}_j; \\ \boldsymbol{B}^{(t)}, \boldsymbol{\Sigma}^{(t)})]. \end{aligned} \tag{8}$$

where, m is the maximal sampling size of $\boldsymbol{\theta}_j$. Here in this model, we sample $\boldsymbol{\gamma}_j$ from multivariate-normal distribution as Equation 2 rather than sampling $\boldsymbol{\theta}_j$ from $p(\boldsymbol{\theta}_j|\boldsymbol{\mathcal{Y}} = \boldsymbol{y}_j, \boldsymbol{x}_j; \boldsymbol{B}^{(t)}, \boldsymbol{\Sigma}^{(t)})$. Then we use $\boldsymbol{\theta}_j = \frac{1}{1+e^{-\boldsymbol{\gamma}_j}}$ to compute $\boldsymbol{\theta}_j$. By combining Equation 2 and 4, we obtain the joint distribution of $(\boldsymbol{\mathcal{Y}} = \boldsymbol{y}_j, \boldsymbol{\theta}_j^{(\tau)})$ given $\boldsymbol{x}_j, \boldsymbol{B}^{(t)}, \boldsymbol{\Sigma}^{(t)}$ as the fol-

lowing equation.

$$
\begin{aligned}
&p(\boldsymbol{\mathcal{Y}} = \boldsymbol{y}_j, \boldsymbol{\theta}_j^{(\tau)} | \boldsymbol{x}_j; \boldsymbol{B}^{(t)}, \boldsymbol{\Sigma}^{(t)}) \\
&= p(\boldsymbol{\mathcal{Y}} = \boldsymbol{y}_j | \boldsymbol{\theta}_j, \boldsymbol{x}_j; \boldsymbol{B}^{(t)}, \boldsymbol{\Sigma}^{(t)}) p(\boldsymbol{\theta}_j | \boldsymbol{x}_j; \boldsymbol{B}^{(t)}, \boldsymbol{\Sigma}^{(t)}) \\
&= \prod_{k=1}^{q} (\theta_{jk})^{y_{jk}-1} (1-\theta_{jk})^{-y_{jk}} \\
&\frac{exp(-\frac{1}{2}(\log \frac{\boldsymbol{\theta}_j}{1-\boldsymbol{\theta}_j} - {\boldsymbol{B}^{(t)}}^T \boldsymbol{x}_j)^T \boldsymbol{\Sigma}^{-1} (\log \frac{\boldsymbol{\theta}_j}{1-\boldsymbol{\theta}_j} - {\boldsymbol{B}^{(t)}}^T \boldsymbol{x}_j))}{(2\pi)^{q/2} |\boldsymbol{\Sigma}^{(t)}|^{1/2}}.
\end{aligned} \tag{9}
$$

4.2. M-step: Maximize Approximate Penalized Expected Log-likelihood

The M-step in this MBLN model is different from the paper (Wu et al., 2018) in three aspects. First, the derivation is different because we use logit as link function. Second, this model uses a simpler technique for implementation. We sample $\boldsymbol{\gamma}$ from a multivariate Gaussian distribution instead of employing Metropolis-hasting algorithm. This reduces the computational cost. Third, the input to estimate $\boldsymbol{B}$ and $\boldsymbol{\Sigma}$ is different. It is caused by the derivation of distinct link function. Next, we will explain these differences.

In the iteration $t+1$ of the MC M-step, we aim to maximize the joint probability of Equation 9. It is equivalent to minimize the average negative log-likelihood $\tilde{Q}'$ in Equation 10.

$$
\begin{aligned}
\tilde{Q}'(\boldsymbol{B}, \boldsymbol{\Sigma} | \boldsymbol{B}^{(t)}, \boldsymbol{\Sigma}^{(t)}) &= -\frac{1}{mn} \sum_{j=1}^{n} \sum_{\tau=1}^{m} [(\log \frac{\boldsymbol{\theta}_j^{(\tau)}}{1-\boldsymbol{\theta}_j^{(\tau)}} \\
&- \boldsymbol{B}^T \boldsymbol{x}_j)^T \boldsymbol{\Sigma}^{-1} (\log \frac{\boldsymbol{\theta}_j^{(\tau)}}{1-\boldsymbol{\theta}_j^{(\tau)}} - \boldsymbol{B}^T \boldsymbol{x}_j) - log|\boldsymbol{\Sigma}^{-1}|].
\end{aligned} \tag{10}
$$

By adding l_1 penalties into the two model parameters, the overall objective function in M-step becomes as Equation 11.

$$
\min_{\boldsymbol{B}, \boldsymbol{\Sigma}^{-1}} \tilde{Q}' + \lambda_1 \|\boldsymbol{B}\|_1 + \lambda_2 \|\boldsymbol{\Sigma}^{-1}\|_1 \tag{11}
$$

Let $\boldsymbol{a}_j = \log \frac{\boldsymbol{\theta}_j}{1-\boldsymbol{\theta}_j} - \boldsymbol{B}^T \boldsymbol{x}_j$. Since $\boldsymbol{a}_j^T \boldsymbol{\Sigma}^{-1} \boldsymbol{a}_j = tr(\boldsymbol{\Sigma}^{-1} \boldsymbol{a}_j \boldsymbol{a}_j^T)$ and $\boldsymbol{a}_j^{(\tau)} = \log \frac{\boldsymbol{\theta}_j^{(\tau)}}{1-\boldsymbol{\theta}_j^{(\tau)}} - \boldsymbol{B}^T \boldsymbol{x}_j$,

$$
\sum_{\tau=1}^{m} {\boldsymbol{a}_j^{(\tau)}}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{a}_j^{(\tau)} = tr(\boldsymbol{\Sigma}^{-1} (\sum_{\tau=1}^{m} {\boldsymbol{a}_j^{(\tau)}}^T \boldsymbol{a}_j^{(\tau)})). \tag{12}
$$

The two model parameters in Equation 11 can be solved by searching a minimal objective value in Equation 13.

$$
\begin{aligned}
\boldsymbol{B}^{(t+1)}, \boldsymbol{\Sigma}^{(t+1)} &= \arg\min_{\boldsymbol{B}, \boldsymbol{\Sigma}^{-1}} \{tr(\boldsymbol{\Sigma}^{-1} (\frac{1}{mn} \sum_{j=1}^{n} \sum_{\tau=1}^{m} {\boldsymbol{a}_j^{(\tau)}}^T \boldsymbol{a}_j^{(\tau)})) \\
&- \log |\boldsymbol{\Sigma}^{-1}| + \lambda_1 \|\boldsymbol{B}\|_1 + \lambda_2 \|\boldsymbol{\Sigma}^{-1}\|_1 \}.
\end{aligned} \tag{13}
$$

This optimization problem in Equation 13 is not a convex problem but has been solved by an iterative algorithm in the paper (Wu et al., 2018). It fixes either $\boldsymbol{B}^{(t)}$ or $\boldsymbol{\Sigma}^{-1}$ in each iteration, then solves another parameter alternatively. We adopt a similar algorithm by supplying different input.

With $\boldsymbol{B}$ fixed at $\boldsymbol{B}^{(t)}$, the optimization problem in Equation 13 turns into a convex optimization problem as shown in Equation 14. We can solve ${\boldsymbol{\Sigma}^{(t+1)}}^{-1}$ by Graphical Lasso (Friedman et al., 2008).

$$
\begin{aligned}
\boldsymbol{\Sigma}^{-1}(\boldsymbol{B}^{(t)}) &= \arg\min_{\boldsymbol{\Sigma}^{-1}} \{tr(\boldsymbol{\Sigma}^{-1} (\frac{1}{mn} \sum_{j=1}^{n} \sum_{\tau=1}^{m} {\boldsymbol{a}_j^{(\tau)}}^T \boldsymbol{a}_j^{(\tau)})) \\
&- \log |\boldsymbol{\Sigma}^{-1}| + \lambda_2 \|\boldsymbol{\Sigma}^{-1}\|_1 \}.
\end{aligned} \tag{14}
$$

The input of Graphical Lasso is an empirical covariance matrix $\boldsymbol{D}$.

$$
\begin{aligned}
\boldsymbol{D} &= \frac{1}{nm} \sum_{j=1}^{n} \sum_{\tau=1}^{m} \boldsymbol{a}_j^{(\tau)} {\boldsymbol{a}_j^{(\tau)}}^T \\
&= \frac{1}{nm} \sum_{j=1}^{n} \sum_{\tau=1}^{m} [\log \frac{\boldsymbol{\theta}_j^{(\tau)}}{1-\boldsymbol{\theta}_j^{(\tau)}} - \boldsymbol{B}^T \boldsymbol{x}_j]^T [\log \frac{\boldsymbol{\theta}_j^{(\tau)}}{1-\boldsymbol{\theta}_j^{(\tau)}} - \boldsymbol{B}^T \boldsymbol{x}_j].
\end{aligned}
$$

When $\boldsymbol{\Sigma}^{-1}$ is fixed at ${\boldsymbol{\Sigma}^{(t)}}^{-1}$, we can estimate $\boldsymbol{B}^{(t+1)}$ from the convex optimization problem presented in Equation 15 by Lasso.

$$
\begin{aligned}
\boldsymbol{B}(\boldsymbol{\Sigma}^{(t)}) &= \arg\min_{\boldsymbol{B}} \{\frac{1}{mn} \sum_{j=1}^{n} \sum_{\tau=1}^{m} ({\boldsymbol{a}_j^{(\tau)}}^T {\boldsymbol{\Sigma}^{(t)}}^{-1} \boldsymbol{a}_j^{(\tau)}) \\
&+ \lambda_1 \|\boldsymbol{B}\|_1 \}.
\end{aligned} \tag{15}
$$

If we write $\boldsymbol{A}$ into the following block matrix.

$$
\boldsymbol{A} = \begin{bmatrix} \log \boldsymbol{\Theta}_1 - \log(1-\boldsymbol{\Theta}_1) - \boldsymbol{X}_1 \boldsymbol{B} \\ \log \boldsymbol{\Theta}_2 - \log(1-\boldsymbol{\Theta}_2) - \boldsymbol{X}_2 \boldsymbol{B} \\ \vdots \\ \log \boldsymbol{\Theta}_n - \log(1-\boldsymbol{\Theta}_n) - \boldsymbol{X}_n \boldsymbol{B} \end{bmatrix}.
$$

where $\boldsymbol{X}_j$ is a $m \times p$ matrix with each row being $\boldsymbol{x}_j$ for all $j = 1, 2, ..., n$. After a series of transformations as shown in Appendix, the objective function in Equation 15 becomes

$\eta(\boldsymbol{B})$ in Equation 16.

$$\eta(\boldsymbol{B}) = \lambda_1 tr(\boldsymbol{B}^{'T}\boldsymbol{B}^{'}) + \frac{1}{mn}\sum_{j=1}^{n} tr((\log\boldsymbol{\Theta}_j - \log(1-\boldsymbol{\Theta}_j) - \boldsymbol{X}_j\boldsymbol{B})^T(\log\boldsymbol{\Theta}_j - \log(1-\boldsymbol{\Theta}_j) - \boldsymbol{X}_j\boldsymbol{B})\boldsymbol{\Sigma}^{(t+1)^{-1}}). \tag{16}$$

where $\hat{\boldsymbol{B}}$ is an estimated coefficient matrix in iteration t, $\boldsymbol{B}^{'} = \boldsymbol{B} \circ \frac{1}{\sqrt{|\hat{\boldsymbol{B}}|}}$, and $\boldsymbol{\Sigma}^{(t+1)^{-1}}$ is the latest computed value in the $t+1$ iteration. Taking the first order derivative of $\eta(\boldsymbol{B})$ w.r.t. $\boldsymbol{B}$ and setting it to zero, and let $(\sum_{j=1}^{n}\boldsymbol{X}_j^T(\log\frac{\boldsymbol{\theta}_j}{1-\boldsymbol{\theta}_j}))\boldsymbol{\Sigma}^{(t+1)^{-1}} = \boldsymbol{H}$ and $\sum_{j=1}^{n}\boldsymbol{X}_j^T\boldsymbol{X}_j = \boldsymbol{S}$, we can solve $\boldsymbol{B}$.

$$vec(\boldsymbol{B}) = [\boldsymbol{\Sigma}^{(t+1)^{-1}} \otimes \boldsymbol{S} + diag(vec(\frac{\lambda_1 mn}{|\hat{\boldsymbol{B}}|}))]^{-1}vec(\boldsymbol{H}). \tag{17}$$

All the detailed derivations are described in the Appendix. Algorithm 1 summarizes this MCEM algorithm. In the E-step, it computes the sum of the log likelihood of the joint probability and penalty on two parameters. In the M-step, It alternatively solves $\boldsymbol{B}$ and $\boldsymbol{\Sigma}^{-1}$ with the other fixed at the value of the latest iteration. When the value in the objective function converges, we get the coefficient matrix $\boldsymbol{B}$ and inverse covariance matrix $\boldsymbol{\Sigma}^{-1}$ in the last iteration.

Algorithm 1 MCEM algorithm

Input: $\boldsymbol{X}$, $\boldsymbol{Y}$,$\boldsymbol{\Sigma}^{(0)^{-1}}$, $\boldsymbol{B}^{(0)}$, λ_1 and λ_2
Output: MLE of $\boldsymbol{B}$ and $\boldsymbol{\Sigma}$
$t = -1$
repeat
  $t = t+1$;
  $\boldsymbol{\gamma}_j \sim \mathcal{N}(\boldsymbol{B}^{(t)^T}\boldsymbol{X}_j, \boldsymbol{\Sigma}^{(t)^2})$;
  $\boldsymbol{a}_j = \boldsymbol{\gamma}_j - \boldsymbol{B}^{(t)^T}\boldsymbol{x}_j$;
  $\hat{\mathcal{Q}}' = -\frac{1}{mn}\sum_{j=1}^{n}\sum_{\tau=1}^{m}[\boldsymbol{a}_j^T\boldsymbol{\Sigma}^{(t)^{-1}}\boldsymbol{a}_j - \log|\boldsymbol{\Sigma}^{(t)^{-1}}|]$;
  $obj = \hat{\mathcal{Q}}' + \lambda_1|\boldsymbol{B}^{(t)}| + \lambda_2|\boldsymbol{\Sigma}^{(t)^{-1}}|$;
  $\boldsymbol{A} = [\boldsymbol{a}_1^{(1)}, \cdots, \boldsymbol{a}_1^{(\tau)}, \cdots, \boldsymbol{a}_1^{(m)}, \cdots, \boldsymbol{a}_n^{(1)}, \cdots, \boldsymbol{a}_n^{(m)}]^T$;
  $\boldsymbol{\Sigma}^{(t+1)^{-1}} = \text{GraphicalLasso}(\boldsymbol{A}, \lambda_2)$;
  $\boldsymbol{S} = \sum_{j=1}^{n}\boldsymbol{X}_j^T\boldsymbol{X}_j$;
  $\boldsymbol{H} = \sum_{j=1}^{n}\boldsymbol{X}_j^T\boldsymbol{\gamma}_j\boldsymbol{\Sigma}^{(t+1)^{-1}}$;
  $\boldsymbol{B}^{(t+1)} = [\boldsymbol{\Sigma}^{(t+1)^{-1}} \otimes \boldsymbol{S} + diag(vec(\frac{\lambda_1 mn}{|\hat{\boldsymbol{B}}|}))]^{-1}vec(\boldsymbol{H})$;
  $\boldsymbol{\Sigma}^{(t)} = \boldsymbol{\Sigma}^{(t+1)}$;
  $\boldsymbol{B}^{(t)} = \boldsymbol{B}^{(t+1)}$;
until converge
return $\{\boldsymbol{B}^{(t+1)}, \boldsymbol{\Sigma}^{(t+1)^{-1}}\}$;

5. WATER TANK DATA STUDY

We use the simulated water tank system dataset (Khorasgani et al., 2019) to demonstrate and validate the performance of our method. This dataset includes a network of water tanks. Each tank can be connected to several tanks in the system. The measurements for each tank includes 1) tank's pressure, which represents the level of water in the tank, 2) tank refill mode, which is equal to 1 when an outside source is filling up the tank, and is 0 otherwise. A tank may start to leak at any point. When the operators fix a leakage, the tank returns to normal operation. The goal is to detect tank's leakages. A leakage in any tank can affect pressure in the tank. However, the leakage is not the only parameter that affects the tank's pressure. The flow-rate between the connected tanks, and the refill flow from an outside source can also affect the pressure. This makes the leak detection problem very challenging. There are totally 100,000 consecutive data points. Each tank has two distinct features Tank Pressure and Tank Refill.

We run this MBLN model on a subset of five connected tanks as shown in Figure 1. The physical structure of these five tanks is an indirected graph. T22 connects to T30 within a distance of 7.10 and T30 bridges with T36, T66, T86 with different distances of 5.25, 10.77 and 5.23. The smaller this distance is, the higher influence of this tank in its neighbor. We aim to predict whether there are leaks for two tanks T20 and T30. Figure 2 describes the data organization for this MBLN model on this water tank data. Instead of only considering the features of itself, MBLN model incorporates the features from each tank's one-hop neighbor tanks. For instance regarding tank T30, we add the features of T22, T36, T66, and T86 besides the features of itself T30. To predict the leak status of two tanks, this model uses all the features from 5 tanks, i.e. 10 features in total. The responses are binary variables for T22 and T30 in parallel. We filter each data point by time. There are four combinations of leak and non-leak for T22 and T30. For example, the first data point has 10 features from T22, T30, T36, T66 and T66. The responses are leaking for both T22 and T30.

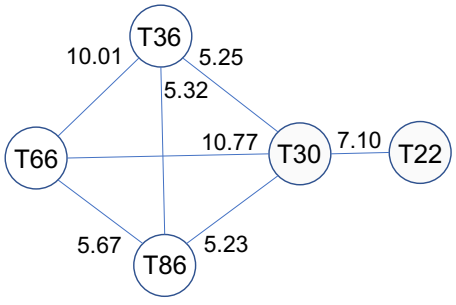

Figure 1. Graph Structure of Water Tank Data.

[0]The dataset is available at https://github.com/IndustrialNetwork/GraphDataset

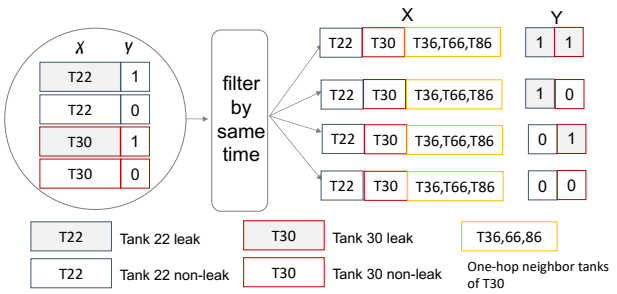

Figure 2. Data Organization on Water Tank Data for MBLN.

We split the dataset into two categories. The first 90% dataset is for training and the left 10% for test. When either T22 or T30 leaks at the same time, this data point is set as leak data. All leak data points from both T22 and T30 are selected in both training and test dataset. In order to balance data for models, we downsample the non-leak data as the same size of leak data. Therefore, there are 37, 499 leak data and equal amount of non-leak data in the training dataset, and 3,830 leak and non-leak data in the test dataset. MBLN model is then applied to predict the leak information of T22 and T30 simultaneously. Other models, such as gradient boost descent, random forest, logistic regression, glmnet and kNN, are used to predict the leak information of T22 and T30 as baselines.

We compare the receiver operating characteristic (ROC) curve of these six models and illustrate them in the Figure 3. It shows that the area under the ROC curve of MBLN model is the largest 0.82. Other approaches, k-nearest neighbor, logistic regression, glmnet, random forest, gradient boost descent have an area of $0.64, 0.70, 0.70, 0.72, 0.79$. MBLN performs the best from the view of ROC curve because its estimated parameters $\hat{\boldsymbol{B}}$ and $\hat{\boldsymbol{\Sigma}}$ reflect the correlation of 5 tanks and contribute to the tank leak prediction of two tanks T22 and T30.

6. CONCLUSION

This paper proposes a multivariate Bernoulli logit-normal model for failure prediction for multiple devices. The insight is that, for devices that are connected and geographically close, there are correlations in the monitoring data collected from these devices. And these correlations can be used to predict failures. We conducted an experiment on a water tank dataset. The prediction results show that this MBLN model is superior than traditional approaches that model each device independently. This MBLN approach for failure prediction has several advantages. First, it can model and predict the failures of multiple devices in a single model so that it predicts failure probabilities for all devices simultaneously. Third, it can deal with both count data and sensor data, or mixed data, as the input of MBLN is normalized before building the model. Last, it can learn the correlation of features

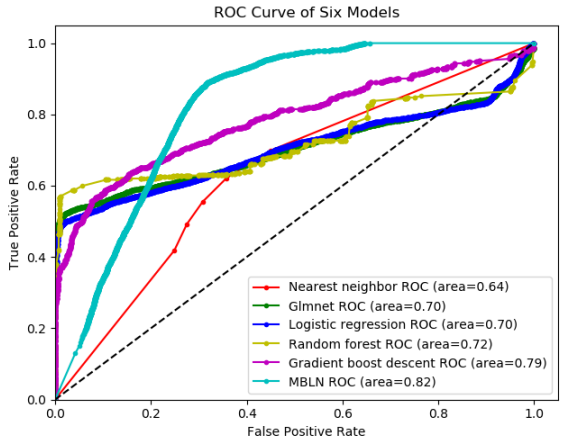

Figure 3. ROC Curve Comparison of Six Models.

from different devices, which can be used by domain experts to learn insights and better understand the behavior of devices.

The scope of this work consists of two main points. One is that MBLN is more effective in handling at least two devices. If only predicting with one device, MBLN becomes similar to glmnet. The other is that the input data from multiple devices should have some correlation. If there's little correlation of input data, the advantages of parameters $\boldsymbol{B}$ and $\boldsymbol{\Sigma}$ can not be embodied.

In future work, we will explore and extend this work to failure predictions for multiple types of devices (i.e., devices with significantly different physical model). Additionally, the proposed MBLN model assumes that the input data are linearly correlated to each other. In the future work, non-linear correlation will be studied.

REFERENCES

Ferrari, R. M., Parisini, T., & Polycarpou, M. M. (2011). Distributed fault detection and isolation of large-scale discrete-time nonlinear systems: An adaptive approximation approach. *IEEE Transactions on Automatic Control*, *57*(2), 275–290.

Friedman, J., Hastie, T., & Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, *9*(3), 432–441.

Gong, P., Ye, J., & Zhang, C. (2012b). Robust multi-task feature learning. In *Proceedings of the 18th acm sigkdd international conference on knowledge discovery and data mining* (pp. 895–903).

Gong, P., Ye, J., & Zhang, C.-s. (2012a). Multi-stage multi-task feature learning. In *Advances in neural information processing systems* (pp. 1988–1996).

Gong, P., Zhou, J., Fan, W., & Ye, J. (2014). Efficient multi-task feature learning with calibration. In *Proceedings of the 20th acm sigkdd international conference on knowledge discovery and data mining* (pp. 761–770).

Jalali, A., Sanghavi, S., Ruan, C., & Ravikumar, P. K. (2010). A dirty model for multi-task learning. In *Advances in neural information processing systems* (pp. 964–972).

Khorasgani, H. (2017). *Model-and data-driven approaches to fault detection and isolation in complex systems* (Unpublished doctoral dissertation).

Khorasgani, H., Farahat, A., Hasanzade, A., & Gupta, C. (2019). Fault detection and isolation in industrial networks using graph convolutional neural networks. In *Ieee phm.*

Khorasgani, H., Farahat, A., Ristovski, K., Gupta, C., & Biswas, G. (2018). A framework for unifying model-based and data-driven fault diagnosis. In *Phm society conference* (Vol. 10).

Liu, H., Wang, L., & Zhao, T. (2014). Multivariate regression with calibration. In *Advances in neural information processing systems* (pp. 127–135).

Lozano, A. C., Jiang, H., & Deng, X. (2013). Robust sparse estimation of multiresponse regression and inverse covariance matrix via the l2 distance. In *Proceedings of the 19th acm sigkdd international conference on knowledge discovery and data mining* (pp. 293–301).

Moon, T. K. (1996). The expectation-maximization algorithm. *IEEE Signal processing magazine*, *13*(6), 47–60.

Ragot, J., & Maquin, D. (2006). Fault measurement detection in an urban water supply network. *Journal of Process Control*, *16*(9), 887–902.

Rothman, A. J., Levina, E., & Zhu, J. (2010). Sparse multivariate regression with covariance estimation. *Journal of Computational and Graphical Statistics*, *19*(4), 947–962.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, *58*(1), 267–288.

Wang, Q., Zheng, S., Farahat, A., Serita, S., & Gupta, C. (2019). Remaining useful life estimation using functional data analysis. *arXiv preprint arXiv:1904.06442*.

Wang, W., Liang, Y., & Xing, E. (2013). Block regularized lasso for multivariate multi-response linear regression. In *Artificial intelligence and statistics* (pp. 608–617).

Wu, H., Deng, X., & Ramakrishnan, N. (2018). Sparse estimation of multivariate poisson log-normal models from count data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, *11*(2), 66–77.

Zheng, S., Ristovski, K., Farahat, A., & Gupta, C. (2017). Long short-term memory network for remaining useful life estimation. In *2017 ieee international conference on prognostics and health management (icphm)* (pp. 88–95).

APPENDIX

M-step

When $\boldsymbol{B}$ is fixed at $\boldsymbol{B}^{(t)}$, we solve $\boldsymbol{\Sigma}^{-1}$ by Graphical Lasso. In order to obtain the covariance matrix $\boldsymbol{D}$ in M-step, we create a virtual vector $\boldsymbol{XX}_j$ to restore $\boldsymbol{D}$.

$$\boldsymbol{XX}_j = [\boldsymbol{a}_j^{(1)}, \cdots, \boldsymbol{a}_j^{(\tau)}, \cdots, \boldsymbol{a}_j^{(m)}]^T. \tag{18}$$

The input matrix $\boldsymbol{XX}$ is an $mn \times q$ matrix , which can be represented as Equation 19.

$$\begin{aligned}\boldsymbol{XX} = [&\boldsymbol{a}_1^{(1)}, \cdots, \boldsymbol{a}_1^{(\tau)}, \cdots, \boldsymbol{a}_1^{(m)}, \cdots, \boldsymbol{a}_j^{(1)}, \cdots, \\ &\boldsymbol{a}_j^{(\tau)}, \cdots, \boldsymbol{a}_j^{(m)}, \cdots, \boldsymbol{a}_n^{(1)}, \cdots, \boldsymbol{a}_n^{(\tau)}, \cdots, \boldsymbol{a}_n^{(m)}]^T.\end{aligned} \tag{19}$$

When $\boldsymbol{\Sigma}^{-1}$ is fixed at ${\boldsymbol{\Sigma}^{(t)}}^{-1}$, we solve $\boldsymbol{B}^{(t+1)}$ in Equation 15 by Lasso.

Suppose $\hat{\boldsymbol{B}}$ is the estimated value of $\boldsymbol{B}$ at iteration t and $1/\sqrt{|\hat{\boldsymbol{B}}|}$ denotes the matrix where each item is the inverse of the square root of the absolute value of the corresponding entry in $\hat{\boldsymbol{B}}$. Then, the l_1 matrix norm penalty in Equation 14 can be approximated as Equation 20.

$$\lambda_1 \|\boldsymbol{B}\|_1 \approx \lambda_1 tr({\boldsymbol{B}'}^T \boldsymbol{B}'). \tag{20}$$

where $\boldsymbol{B}' = \boldsymbol{B} \circ \frac{1}{\sqrt{|\hat{\boldsymbol{B}}|}}$. Here, $\circ$ represents the Hadamard (element-wise) product.

The objective function is as Equation 16. Now we try to obtain the derivation of the first term of Equation 16.

$$\frac{\partial {\boldsymbol{B}'}^T \boldsymbol{B}'}{\partial \boldsymbol{B}} = (\frac{1}{\sqrt{|\hat{\boldsymbol{B}}|}})^T \cdot (\boldsymbol{B} \circ \frac{1}{\sqrt{|\hat{\boldsymbol{B}}|}}) + (\boldsymbol{B} \circ \frac{1}{\sqrt{|\hat{\boldsymbol{B}}|}})^T \cdot \frac{1}{\sqrt{|\hat{\boldsymbol{B}}|}}.$$

According to the trace rule operations $tr(A^T) = tr(A)$ and $tr(AB) = tr(BA)$, we can derive that

$$tr(\frac{\partial {\boldsymbol{B}'}^T \boldsymbol{B}'}{\partial \boldsymbol{B}}) = 2\boldsymbol{B} \circ \frac{1}{|\hat{\boldsymbol{B}}|}. \tag{21}$$

Furthermore, let $\boldsymbol{\Omega}^{(t+1)} = {\boldsymbol{\Sigma}^{(t+1)}}^{-1}$, the derivation of the second term of Equation 16 is as follows.

$$\begin{aligned}&\frac{\partial[(\log \frac{\boldsymbol{\theta}_j}{1-\boldsymbol{\theta}_j} - \boldsymbol{X}_j\boldsymbol{B})^T (\log \frac{\boldsymbol{\theta}_j}{(1-\boldsymbol{\theta}_j)} - \boldsymbol{X}_j\boldsymbol{B})\Omega^{(t+1)}]}{\partial \boldsymbol{B}} \\ &= -{\boldsymbol{X}_j}^T \log \frac{\boldsymbol{\theta}_j}{(1-\boldsymbol{\theta}_j)} \Omega^{(t+1)} + \boldsymbol{X}_j^T \boldsymbol{X}_j \boldsymbol{B} \Omega^{(t+1)} \\ &- (\log \frac{\boldsymbol{\theta}_j}{1-\boldsymbol{\theta}_j})^T \boldsymbol{X}_j \Omega^{(t+1)} + \boldsymbol{B}^T \boldsymbol{X}_j^T \boldsymbol{X}_j \Omega^{(t+1)}.\end{aligned} \tag{22}$$

Since $tr(\boldsymbol{X}_j^T \boldsymbol{X}_j \boldsymbol{B}) = tr(\boldsymbol{B}^T \boldsymbol{X}_j^T \boldsymbol{X}_j)$ and $tr({\boldsymbol{X}_j}^T \log \frac{\boldsymbol{\theta}_j}{(1-\boldsymbol{\theta}_j)}) = tr((\log \frac{\boldsymbol{\theta}_j}{1-\boldsymbol{\theta}_j})^T \boldsymbol{X}_j)$, we have the derivation of objective function as shown in Equation 23.

$$\frac{\partial \eta(\boldsymbol{B})}{\partial \boldsymbol{B}} = 2\lambda_1(\boldsymbol{B} \circ \frac{1}{|\hat{\boldsymbol{B}}|}) + 2\frac{1}{mn}\sum_{j=1}^{n}[\boldsymbol{X}_j^T\boldsymbol{X}_j\boldsymbol{B}\Omega^{(t+1)} - \boldsymbol{X}_j^T \log\frac{\boldsymbol{\theta}_j}{1-\boldsymbol{\theta}_j}\Omega^{(t+1)}]. \quad (23)$$

Taking the first order derivative of $\eta(\boldsymbol{B})$ w.r.t. $\boldsymbol{B}$ and setting it to zero, we have Equation 24.

$$(\sum_{j=1}^{n}\boldsymbol{X}_j^T\boldsymbol{X}_j)\boldsymbol{B}\Omega^{(t+1)} + \boldsymbol{B}\circ\frac{\lambda_1 mn}{|\hat{\boldsymbol{B}}|} = \sum_{j=1}^{n}(\boldsymbol{X}_j^T(\log\frac{\boldsymbol{\theta}_j}{1-\boldsymbol{\theta}_j}))\Omega^{(t+1)}. \quad (24)$$

If we let $(\sum_{j=1}^{n}\boldsymbol{X}_j^T(\log\frac{\boldsymbol{\theta}_j}{1-\boldsymbol{\theta}_j}))\boldsymbol{\Omega}^{(t+1)} = \boldsymbol{H}$ and $\sum_{j=1}^{n}\boldsymbol{X}_j^T\boldsymbol{X}_j = \boldsymbol{S}$, and apply the matrix vectorization operator $vec(\cdot)$ to both sides of Equation 24, we have

$$(\Omega^{(t+1)T}\otimes\boldsymbol{S})vec(\boldsymbol{B}) + vec(\frac{\lambda_1 mn}{|\hat{\boldsymbol{B}}|})\circ vec(\boldsymbol{B}) = vec(\boldsymbol{H}). \quad (25)$$

Here $\otimes$ represents the Kronecker product. By pulling out $vec(\boldsymbol{B})$ to the left side of the equation, we have Equation 26.

$$vec(\boldsymbol{B}) = [\Omega^{(t+1)}\otimes\boldsymbol{S} + diag(vec(\frac{\lambda_1 mn}{|\hat{\boldsymbol{B}}|}))]^{-1}vec(\boldsymbol{H}). \quad (26)$$

The estimated coefficient matrix $\boldsymbol{B}$ can be attained via reshaping $vec(\boldsymbol{B})$.