Structural Break Detection in Non-Stationary Network Vector Autoregression Models

Yi Han and Thomas C. M. Lee, *Senior Member, IEEE*

***Abstract*—Imagine a network, like a socialnetwork or a system of connected devices, is being observed over time. Each node in this network has certain measurements attached to it that can change, like the temperature of a device. Although the overall structure of the network remains constant, these measurements can vary, leading to a complex multivariate time series dataset that exhibits non-stationary characteristics over time. This paper applies a piecewise stationary network vector autoregressive (NAR) model to analyze these network data. The main idea is to partition the entire dataset into segments where the NAR model for each segment remains stationary. The identification of these segments, along with the determination of the NAR processes' autoregressive lag orders, are treated as unknowns. The minimum description length (MDL) principle is employed to develop a criterion for model selection that estimates these unknown parameters. A two-stage genetic algorithm is then formulated to tackle this optimization challenge. The MDL criterion is proven to be consistent in identifying the number and positions of the breakpoints - the junctures where adjacent NAR segments intersect. The effectiveness of the proposed method is demonstrated through simulation studies and real data analysis.**

***Index Terms*—Breakpoint, changepoint, genetic algorithms, minimum description length (MDL), piecewise network vector autoregressive (NAR) model.**

I. Introduction

CONSIDER a network $A = (a_{i_1 i_2}) \in \mathbb{R}^{K\times K}$ with K nodes that may represent different relationships in different situations, such as people's social networks, companies' economic networks, and physical site networks. Let $a_{i_1 i_2} = 1$ if there exists some kind of relationship from node i_1 to node i_2; for example, followers and followees on social media. On the other hand, $a_{i_1 i_2} = 0$ if such a relationship does not exist. Also, further assume A cannot be self-related: $a_{i_l i_l} = 0$ for $i_l = 1, \ldots, K$. Such relationships can be either directed or undirected.

From the network A we can collect continuous measurements $X_{it} \in \mathbb{R}$ from node $i = 1, \ldots, K$ at time $t = 1, \ldots, T$. Denote

$$\boldsymbol{X}_t = (X_{1t}, \ldots, X_{Kt})^T \in \mathbb{R}^K, \quad t = 1, \ldots, T,$$

as the measurements from all K nodes in the whole network at time point t.

One of the earlier and widely used models for the $\boldsymbol{X}_t$'s is the vector autoregressive (VAR) model; e.g., see [1] and [2]. The VAR model introduces $O(K^2)$ parameters to handle the interactions amongst the nodes, but the estimation problem is tremendously large if K is large. Besides, there might be other exogenous covariates related to the nodes that also influence the $\boldsymbol{X}_t$'s; e.g., personal information in social networks and regional development level in economic networks. The VAR model unusually fails to include such information.

The network vector autoregressive (NAR) model was thus proposed by [3] to model the $\boldsymbol{X}_t$'s. It contains much fewer parameters that also utilize the observed network structure A and also allows possible exogenous covariates. Other time series models designed for networks include [4], [5].

For each node i, assume there exists a q dimensional node-specific exogenous covariates $\boldsymbol{V}_i = (V_{i1}, \ldots, V_{iq})^T \in \mathbb{R}^q$. As stated in [3] and [6], a NAR(p_1, p_2) model assumes the measurements X_{it}'s are influenced by self lags (past values), network lags (past values of "related" nodes), and node specified covariates effects, and is given by

$$X_{it} = \beta_0 + \boldsymbol{V}_i^\top \boldsymbol{\gamma} + \sum_{m=1}^{p_1} \alpha_m \sum_{j=1}^{K} \frac{a_{ij}}{n_i} X_{j(t-m)} + \sum_{n=1}^{p_2} \beta_n X_{i(t-n)} + \varepsilon_{it}, \tag{1}$$

where $n_i = \sum_{l\neq i} a_{il}$ is the total number of nodes that i follows, $\beta_0, \alpha_m \in \mathbb{R}$, $\beta_n \in \mathbb{R}$, and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_q) \in \mathbb{R}^q$ are, respectively, the coefficients for the network lags, the self lags and the node specified covariates. Also, p_1 and p_2 are the lag orders for the network lags and the self lags, respectively. The noise ε_{it} is assumed to follow a normal distribution $N(0, \sigma_i^2)$. Lastly, write $\boldsymbol{W} = \text{diag}\{n_1^{-1}, \ldots, n_K^{-1}\}\boldsymbol{A} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k)^T$ as the row-normalized network.

The NAR model has been successfully applied to solve problems in different areas, including social media analysis [3], air quality studies [6], and economic growth evaluations [7]. However, the vast majority of these studies assume the underlying process is stationary over the whole time span, which can be an unrealistic assumption for multivariate time series observed in many modern applications [8], [9].

One possible approach to mitigate this issue is to partition the whole process into a number of shorter, stationary processes.

Manuscript received 17 September 2023; revised 28 March 2024; accepted 3 May 2024. Date of publication 9 May 2024; date of current version 16 August 2024. This work was supported in part by the National Science Foundation under Grant CCF-1934568, Grant DMS-1916125, Grant DMS-2113605, and Grant DMS-2210388. Recommended for acceptance by Prof. C Gao. *(Corresponding author: Thomas C. M. Lee.)*

The authors are with the Department of Statistics, University of California, Davis, Davis, CA 95616 USA (e-mail: yyihan@ucdavis.edu; tcmlee@ucdavis.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TNSE.2024.3398002, provided by the authors.

Digital Object Identifier 10.1109/TNSE.2024.3398002

2327-4697 © 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Univ of Calif Davis. Downloaded on August 23,2024 at 16:40:49 UTC from IEEE Xplore. Restrictions apply.

That is, a sequence of piecewise stationary NAR models is used to model the non-stationary series $\{\boldsymbol{X}_t\}_{t=1}^T$.

A precise formulation is as follows. Suppose there are m_0 breakpoints; i.e., $\{\boldsymbol{X}_t\}_{t=1}^T$ is partitioned into m_0+1 piecewise stationary NAR models. The m_0 breakpoint locations $\{\tau_j\}_{j=1}^{m_0}$ satisfy $0<\tau_1<\tau_2<\ldots<\tau_{m_0}<T+1$, and for convenience, write $0=\tau_0$ and $\tau_{m_0+1}=T+1$. For all $j=1,\ldots,m_0+1$, it is assumed that the j-th segment, $\{\boldsymbol{X}_t\}$ with $\tau_{j-1}\le t<\tau_j$, follows a stationary NAR$(p_{1,j},p_{2,j})$ model. It is also assumed that the network structure remains unchanged in all segments. Similar to (1), the j-th segment is modeled as

$$X_{it,j}=\beta_{0,j}+\boldsymbol{V}_i^\top\boldsymbol{\gamma}_j+\sum_{m=1}^{p_{1,j}}\alpha_{m,j}\sum_{l=1}^{K}\frac{a_{il}}{n_i}X_{l(t-m)}+\sum_{n=1}^{p_{2,j}}\beta_{n,j}X_{i(t-n)}+\varepsilon_{it,j}. \tag{2}$$

Throughout the paper, we follow the same stationarity assumptions in [3] for each segment, for example, $\sum_{i=1}^{p_{1,j}}(|\alpha_i|+|\beta_i|)<=1$ is satisfied which guarantees the piecewise stationarity. With the above piecewise NAR model, one needs to estimate the number m_0 and the locations $\{\tau_1,\ldots,\tau_{m_0}\}$ of the breakpoints. One also needs to estimate the model parameters in each segment, including the lag orders $p_{1,j}$ and $p_{2,j}$, and the regression coefficients $\beta_{0,j}$, $\alpha_{m,j}$, and $\beta_{n,j}$. It will be shown below that for each segment, once the lag orders are determined, the regression coefficient estimates can be obtained using maximum likelihood [3]. So the main challenge is to estimate the number and locations of the breakpoints, as well as the lag orders in each segment. Notice that this can be seen as a statistical model selection problem, as different m_0 would lead to different piecewise stationary models with different numbers of model parameters.

One major contribution of this paper is the development of a systematic method for selecting a best-fitting piecewise stationary NAR model (2). That is, to estimate the number and locations of the breakpoints, as well as the orders for each stationary NAR model between any two adjacent breakpoints. Once these quantities are estimated, the remaining model parameters can be estimated using maximum likelihood. The proposed method invokes the minimum description length (MDL) principle [10], [11] to derive an objective criterion for model selection, and uses the genetic algorithm to solve the corresponding optimization problem.

Breakpoint detection in network problems has been widely investigated in recent years. Existing mainstream methods can be broadly divided into two categories. The first group of methods begins with summarizing a certain characteristic of each of the networks with a metric and then detects any possible breakpoints with respect to that metric. Examples of a network metric include various matrix norms [12], [13] and centrality metrics [14], [15]. Reducing a (complicated) network to a simple metric typically provides substantial speed gain, but at the same time, it may inevitably cause information loss, which in turn may adversely affect the final results. The second group of methods fits a dynamic network model to the data and uses model-based testing methods to detect breakpoints. Examples of such a network model include generalized hierarchical random graphs [16], Kronecker product graphs [17], and stochastic block models [18]. Some of these model assumptions could be restrictive, but if appropriate for the data at hand, these methods tend to provide excellent results.

One merit of the proposed method is that no strong restrictions are imposed on the network structures, which greatly increases the applicability of the method. To the best of the authors' knowledge, this is one of the first complete systematic studies that consider structural break estimation in non-stationary NAR models.

The rest of this paper is organized as follows. Section II derives the MDL criterion for estimating the unknowns in the piecewise stationary NAR model. It also studies the theoretical properties of the criterion. Section III develops a two-stage GA algorithm to minimize the MDL criterion. The empirical performance of the proposed method is illustrated in Section IV via various numerical simulations and in Section V via an application to some real Manhattan yellow cab data. Lastly, concluding remarks are offered in Section VI, while technical details and additional simulation results are provided in the supplementary material.

II. Breakpoint Detection Using MDL

The MDL principle is a popular method for deriving an effective model selection criterion. It defines the best-fitting model as the one that compresses the data into the shortest possible code length for storage, where the code length represents the bites needed to store the data. It was proposed by Rissanen [10], [11] and has been successfully applied to solve various model selection problems such as image segmentation [19], network constructions [20], [21], [22], and quantile and spline regression [23], [24]. This paper focuses on the so-called "Two–Part MDL" [25], and this section derives the corresponding MDL criterion for fitting a piecewise stationary NAR model.

A. Derivation of the MDL Criterion

To store the observed data, one can split them into two parts: the first part is a fitted model and the second part is the corresponding residuals. If the fitted model is a good model, it will be more economical to store the data in this way. Denote CL(z) as the code length of any object z; thus, we want to minimize CL("data"). Also, denote the whole class of piecewise NAR models as $\mathcal{M}$, denote any model in $\mathcal{M}$ as $\mathcal{F}\in\mathcal{M}$ and its corresponding residuals as $\hat{\mathcal{E}}$. Then we have

$$\begin{aligned}\text{CL("data")}&=\text{CL("fitted model")}+\text{CL("residuals")}\\&=\text{CL}(\mathcal{F})+\text{CL}(\hat{\mathcal{E}}|\mathcal{F}).\end{aligned} \tag{3}$$

We need a computable expression for CL("data") and we first calculate CL$(\mathcal{F})$. Notice that to completely specify a model $\mathcal{F}$, we need to know the breakpoint number m and their locations $\mathcal{T}=\{\tau_1,\ldots,\tau_m\}$. In addition, for all $j=1,\ldots,m+1$, we need to know the lag orders $\boldsymbol{p}_j=(p_{1,j},p_{2,j})$ and regression parameters $\boldsymbol{\theta}_j=(\beta_{0,j},\alpha_{1,j},\ldots,\alpha_{p_{1,j},j},\beta_{1,j},\ldots,\beta_{p_{2,j},j},\boldsymbol{\gamma}_j)$, for the j-th segment. Write $\mathcal{P}=(\boldsymbol{p}_1,\ldots,\boldsymbol{p}_{m+1})$ and $\hat{\Theta}=(\hat{\boldsymbol{\theta}}_1,\ldots,\hat{\boldsymbol{\theta}}_{m+1})$. Then we have $\mathcal{F}=(m,\mathcal{T},\mathcal{P},\hat{\Theta})$, which leads

Authorized licensed use limited to: Univ of Calif Davis. Downloaded on August 23,2024 at 16:40:49 UTC from IEEE Xplore. Restrictions apply.

to the code length decomposition:

$$\mathrm{CL}(\mathcal{F}) = \mathrm{CL}(m) + \mathrm{CL}(\mathcal{T}) + \mathrm{CL}(\mathcal{P}) + \mathrm{CL}(\hat{\Theta}). \tag{4}$$

When applying MDL, the code length of an unknown positive integer I can be approximated by $\log(I)$ [10]. On the other hand, if I is known to be upper-bounded by I_u, then its code length is $\log(I_u)$. So the first three terms on the RHS of (4) are

$$\mathrm{CL}(m) = \log(m+1), \tag{5}$$

$$\mathrm{CL}(\mathcal{T}) = (m+1)\log(T), \tag{6}$$

$$\mathrm{CL}(\mathcal{P}) = \sum_{j=1}^{m+1} \{\log(p_{1,j}) + \log(p_{2,j})\}, \tag{7}$$

where the additional 1 in $\mathrm{CL}(m)$ is used to make the formula meaningful when $m = 0$.

For the last term in (4), we need to first estimate $\hat{\Theta}$ from model (2) and then encode the resulting estimated values. For estimation, we shall use the maximum likelihood method of [3], while for encoding, we shall use the result of [10] that any (scalar) maximum likelihood estimate calculated from N observations can be effectively encoded with $\frac{1}{2}\log(N)$ bits. We first describe the maximum likelihood method of [3].

Let $\boldsymbol{w}_i = (a_{il}/n_i : 1 \le l \le K)^T \in \mathbb{R}^K$ be the i-th row vector of the row normalized network matrix $\boldsymbol{W}$, and

$$Z^*_{l(t-1),j} := \{1, \boldsymbol{w}_l^T \boldsymbol{X}_{t-1,j}, \ldots, \boldsymbol{w}_l^T \boldsymbol{X}_{t-p_{1,j},j}, X_{l(t-1),j}, \ldots, X_{l(t-p_{2,j}),j}, \boldsymbol{V}_l^T\}^T \in \mathbb{R}^{p_{1,j}+p_{2,j}+q+1},$$

where $X_{l(t-1),j}$ represents the l-th element of $\boldsymbol{X}_{t-1,j}$. Let

$$\boldsymbol{Z}^*_{t-1,j} := (Z^*_{1(t-1),j}, \ldots, Z^*_{K(t-1),j})^T \in \mathbb{R}^{K\times(p_{1,j}+p_{2,j}+q+1)}.$$

Then the j-th segment, which is a NAR$(p_{1,j}, p_{2,j})$ model (see (2)), can be rewritten in vector form as

$$\boldsymbol{X}_{t,j} = \boldsymbol{Z}^*_{t-1,j}\boldsymbol{\theta}_j + \boldsymbol{\varepsilon}_j, \tag{8}$$

where $\boldsymbol{\varepsilon}_j \sim N_k(\boldsymbol{0}, \sigma_j^2 \boldsymbol{I}_k)$. Here the variances do not need to be the same for the proposed method to work, but for simplicity, below we will assume they are identical. With this, the maximum likelihood estimator of $\boldsymbol{\theta}_j$ is

$$\hat{\boldsymbol{\theta}}_j = \left(\sum_{t=\tau_{j-1}+p_{\max,j}+1}^{\tau_j} \boldsymbol{Z}^{*T}_{t-1,j}\boldsymbol{Z}^*_{t-1,j}\right)^{-1} \times \sum_{t=\tau_{j-1}+p_{\max,j}+1}^{\tau_j} \boldsymbol{Z}^*_{t-1,j}\boldsymbol{X}_{t,j} \in \mathbb{R}^{(p_{1,j}+p_{2,j}+q+1)}, \tag{9}$$

where $p_{\max,j} := \max(p_{1,j}, p_{2,j})$, $n_j := \tau_j - \tau_{j-1}$, and

$$\hat{\sigma}_j^2 = \frac{\sum_{t=\tau_{j-1}+p_{\max,j}+1}^{\tau_j} (\boldsymbol{X}_{t,j} - \boldsymbol{Z}^*_{t-1,j}\hat{\boldsymbol{\theta}}_j)^T(\boldsymbol{X}_{t,j} - \boldsymbol{Z}^*_{t-1,j}\hat{\boldsymbol{\theta}}_j)}{K(n_j - p_{\max,j})}. \tag{10}$$

As mentioned before, to encode a scalar maximum likelihood estimate, the code length is $\frac{1}{2}\log(N)$ if N observations were used for estimation. Therefore,

$$\mathrm{CL}(\hat{\Theta}) = \sum_{j=1}^{m+1} \frac{p_{1,j}+p_{2,j}+q+1}{2}\log(n_j). \tag{11}$$

The last term in (3) that we need to calculate is $\mathrm{CL}(\hat{\mathcal{E}}|\mathcal{F})$, which equals the negative log (base 2) of the likelihood of the fitted model $\mathcal{F}$ [10]. From (8), (9), and (10), we have

$$\mathrm{CL}(\hat{\mathcal{E}}|\mathcal{F}) = \sum_{j=1}^{m+1}\left[\frac{K(n_j - p_{\max,j})}{2}\left\{\log(2\pi\hat{\sigma}_j^2)+1\right\}\right]\log_2 e \tag{12}$$

Combining (5), (6), (7), (11), and (12) and using logarithm base e instead of base 2, (3) becomes

$$\begin{aligned}
&\mathrm{CL}(\text{“data”}) = \log(m+1) + (m+1)\log(T) \\
&+ \sum_{j=1}^{m+1}\left(\log(p_{1,j}) + \log(p_{2,j}) + \frac{p_{1,j}+p_{2,j}+q+1}{2}\log(n_j)\right) \\
&+ \sum_{j=1}^{m+1}\left\{\frac{K(n_j - p_{\max,j})}{2}\left(\log(2\pi\hat{\sigma}_j^2)+1\right)\right\}\log_2 e \\
&:= \mathrm{MDL}(m, \tau_1, \ldots, \tau_m, p_{1,1}, p_{2,1}, \ldots, p_{1,m+1}, p_{2,m+1}).
\end{aligned} \tag{13}$$

Thus, the MDL principle suggests that the best-fitting model for the observed data $\boldsymbol{X}_{t,j}, t = 1, \ldots, n_j, j = 1, \ldots, m$ is the one $\mathcal{F} \in \mathcal{M}$ that minimizes (13).

B. Theoretical Properties

Denote the true number of breakpoints as m_0 and the true locations of the breakpoints as $\mathcal{T}_0 = \{\tau_1^0, \ldots, \tau_{m_0}^0\}$. Define the true relative breakpoint locations as $\boldsymbol{\lambda}_0 = \{\lambda_1^0, \ldots, \lambda_{m_0}^0\}$ with $\tau_j^0 = \lfloor \lambda_j^0 T \rfloor$ for $j = 1, \ldots, m_0$, where $\lfloor x \rfloor$ represents the greatest integer that is less than or equal to x. Further, write $\boldsymbol{p} = (p_{1,1}, p_{2,1}, \ldots, p_{1,m+1}, p_{2,m+1})$ and $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m)$. Note that the theoretical results in this subsection will be presented in terms of $\boldsymbol{\lambda}$ instead of $\mathcal{T}$.

As suggested by [26], for each segment, a sufficient number of data points are required to adequately estimate the corresponding NAR model parameters. For this reason, we impose the following constraint on the estimate of $\boldsymbol{\lambda}$. First, choose $\xi > 0$ sufficiently small enough that $\xi \ll \min_{i=1,\ldots,m_0+1}(\lambda_i^0 - \lambda_{i-1}^0)$. Then define

$$\begin{aligned}
A_m = \{&(\lambda_1, \ldots, \lambda_m \\
&0 = \lambda_0 < \lambda_1 < \cdots < \lambda_m < \lambda_{m+1} = 1, \\
&\lambda_i - \lambda_{i-1} \ge \xi, i = 1, 2, \ldots, m+1\}
\end{aligned}$$

Lastly, we require the estimate of $\boldsymbol{\lambda}$ to be an element of A_m.

Using this constraint and (13), the unknown meta-parameters are given by

$$\{\hat{m}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{p}}\} = \arg\min_{m,\boldsymbol{p},\boldsymbol{\lambda}\in A_m} \frac{2}{T}\mathrm{MDL}(m, \boldsymbol{\lambda}, \boldsymbol{p}). \tag{14}$$

Authorized licensed use limited to: Univ of Calif Davis. Downloaded on August 23,2024 at 16:40:49 UTC from IEEE Xplore. Restrictions apply.

Theorem 2.1: For the piecewise stationary NAR model given by (2), when the true number of breakpoints m_0 is known, the estimate $\hat{\boldsymbol{\lambda}}$ defined by (14) satisfies

$$\hat{\lambda}_j \xrightarrow{a.s.} \lambda_j^0, \quad j = 1, \ldots, m_0.$$

Corollary 2.1.1: If the number of breakpoints m_0 is unknown and estimated with (14), then

1) The estimated number of breakpoints $\hat{m} \geq m_0$ for sufficient large T.
2) When $\hat{m} > m_0$, for any $\lambda_j^0 \in \boldsymbol{\lambda}_0$, there exists a $\hat{\lambda}_k$ such that $|\lambda_j^0 - \hat{\lambda}_k| < \epsilon, \forall \epsilon > 0$ for large enough T.
3) The lag order of the model in each segment cannot be underestimated; i.e., $\hat{p}_{1,j} \geq p_{1,j}^0$, $\hat{p}_{2,j} \geq p_{2,j}^0$, where $p_{1,j}^0$ and $p_{2,j}^0$ are the true lag orders.

If Assumption 1 below is satisfied, a consistency result of the MDL estimator (14) can be derived even when m_0 is not known.

Assumption 1: For $j = 1, \ldots, m+1$, any fixed p_j and any sequence $\{g(T)\}_{T \leq 1}$ of integers that satisfies $g(T) \leq cT^{0.5}$ for some $c > 0$ when T is sufficiently large. Let $f_{\boldsymbol{p_j}}(\boldsymbol{X}_{i,j} | \boldsymbol{X}_{s,j}, s < i; \boldsymbol{\theta}_j)$ be the conditional density function of the i-th observation in the j-th segment. Also let $l_j(\boldsymbol{p}_j, \boldsymbol{\theta}_j, \boldsymbol{X}_{i,j} | \boldsymbol{X}_{s,j}, s < i) = \log f_{\boldsymbol{p}_j}(\boldsymbol{X}_{i,j} | \boldsymbol{X}_{s,j}, s < i; \boldsymbol{\theta_j})$ be the conditional log-likelihood function for $\boldsymbol{X}_{i,j}$. then

$$\frac{1}{g(T)} \sum_{i=T-g(T)+1}^{T} l_j(\boldsymbol{p}_j, \boldsymbol{\theta}_j, \boldsymbol{X}_{i,j} | \boldsymbol{X}_{s,j}, s < i) \xrightarrow{a.s.} E(l_j(\boldsymbol{p}_j, \boldsymbol{\theta}_j, \boldsymbol{X}_{1,j} | \boldsymbol{X}_{s,j}, s < 1))$$

and

$$\frac{1}{g(T)} \sum_{i=T-g(T)+1}^{T} l_j'(\boldsymbol{p}_j, \boldsymbol{\theta}_j, \boldsymbol{X}_{i,j} | \boldsymbol{X}_{s,j}, s < i) \xrightarrow{a.s.} E(l_j'(\boldsymbol{p}_j, \boldsymbol{\theta}_j, \boldsymbol{X}_{1,j} | \boldsymbol{X}_{s,j}, s < 1))$$

where $l_j'(\boldsymbol{p}_j, \boldsymbol{\theta}_j, \boldsymbol{X}_{i,j} | \boldsymbol{X}_{s,j}, s < i)$ is the first derivative of $l_j(\boldsymbol{p}_j, \boldsymbol{\theta}_j, \boldsymbol{X}_{i,j} | \boldsymbol{X}_{s,j}, s < i)$.

This assumption is needed to control the effects at the two ends of the fitted segments so that the convergence rate of the location estimator can be established.

Theorem 2.2: For the piecewise stationary NAR model given by (2), under the assumptions of Theorem II.1(except for the known number of breakpoints) and Assumption 1, the estimator $\{\hat{m}, \hat{\boldsymbol{\lambda}}\}$ defined by (14) satisfies

$$\hat{m} \xrightarrow{a.s.} m_0, \quad \hat{\boldsymbol{\lambda}} \xrightarrow{a.s.} \boldsymbol{\lambda^0}.$$

The proofs of Theorem II.1, Corollary II.1.1, and Theorem II.2 can be found in the supplementary material.

III. Practical Optimization of MDL Using Genetic Algorithms

The enormous searching space makes the minimization of (13) or (14) a non-trivial task. This section develops a genetic algorithm (GA) for solving this problem.

A. A Brief Introduction to Genetic Algorithms

GA is a search heuristic that can be dated back as early as [27], for which the main idea was inspired by Charles Darwin's theory of natural evolution. Typically, a GA begins with generating an initial set of possible solutions (*chromosomes*) to the optimization problem of interest, which is represented by vector form. Next, these chromosomes are weighted sampled as parents to generate their "offspring": parent chromosomes with better values for the optimization problem (i.e., larger values for maximization problems or smaller values for minimization problems) have higher chances of being chosen. An offspring chromosome is then produced by applying either a *crossover* or a *mutation* operation to the chosen parent chromosomes. Such a process repeats until some stopping criteria are met.

As suggested in [26], to preserve the evolution direction towards the optimal value, the best chromosome from the previous generation is preserved to replace the worst chromosome of the current generation. This process is known as the *elitist* step and guarantees the monotonously of the algorithm.

To speed up the algorithm, [28] introduced an island model version that is particularly suited for parallel computing. Rather than running with only one group of evolving chromosomes, the island model can simultaneously run NI (number of islands) subgroups of chromosomes. Periodically, chromosomes are allowed to migrate amongst the islands, a process known as *migration*. The migration policy that we use here is the same as in [26]. The purpose of migration is to avoid sub-optimal solutions for the subgroups. At every M_i-th generation, the worst M_N chromosomes in j-th island are replaced by the best M_N chromosomes in $(j-1)$-th island, for $j = 2, \ldots, NI$. The first island's worst M_N chromosomes are replaced by the best M_N chromosomes in the NI-th island.

B. Implementation Details

This subsection provides details of the tailored GA that we use to minimize (13) for the piecewise stationary NAR model (2).

1) Chromosome Representation: In general, the representation of chromosomes plays an important role in the overall performance of GAs. A good representation should contain all the needed information of any potential solution for the calculation of (13). For the current problem, it suffices to include only the breakpoints $\mathcal{T}$ and the lag orders $\boldsymbol{p}$, as once these quantities are specified, the remaining unknown parameters can be uniquely calculated. Given this, we propose using the following constant-length representation for a chromosome $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_T)$, where the gene values are

$$\begin{cases} \delta_t = -1 & \text{if time t is not a breakpoint} \\ \delta_t = (p_{1,j}, p_{2,j}) & \text{if } t = \tau_{j-1} \text{ (i.e., time t is the (j-1)-th breakpoint) and the j-th segment is a NAR } (p_{1,j}, p_{2,j}) \text{ model.} \end{cases} \tag{15}$$

If t is a breakpoint location, the t-th gene consists of two values, even though together they only use one gene index. This allows the length of the chromosomes to remain constant T irrespective

Authorized licensed use limited to: Univ of Calif Davis. Downloaded on August 23,2024 at 16:40:49 UTC from IEEE Xplore. Restrictions apply.

TABLE I
MINIMUM NUMBER OF DATA POINTS REQUIRED FOR DIFFERENT $p_{\max}$

$p_{\max}$	0-1	2	3	4	5	6	7	8	9-10	11-20
$m_{p_{\max}}$	10	12	14	16	18	20	22	24	25	50

Algorithm 1: Initialization.

Input: Minimum span for different lag order: $m_{p_{\max}}$;
Probability of being a breakpoint: r_G;
The upper bound of lag orders: P_0;
Output: Initialization chromosome δ
initialize $\delta = (\delta_1, \cdots, \delta_T) = (-1, \cdot, -1)$; $i = 1$
while $i \leq T$ **do**
  Generate $r \sim Uniform(0, 1)$
  if $i == 1$ **then**
    SAMPLE $p_{1,i}, p_{2,i} \sim [P_0]$ SET $\delta_i = (p_{1,i}, p_{2,i})$; $p_{\max,i} = \max(p_{1,i}, p_{2,i})$ $i = i + m_{p_{\max,i}} + 1$
  else
    if $r < r_G$ **then**
      SAMPLE $p_{1,i}, p_{2,i} \sim [P_0]$ SET $\delta_i = (p_{1,i}, p_{2,i})$; $p_{\max,i} = \max(p_{1,i}, p_{2,i})$ $i = i + m_{p_{\max,i}} + 1$
    else
      $i = i + 1$
    end
  end
end

of the number of breakpoints, which in turn facilitates the execution of the crossover and mutation operations.

2) Maximum Lag Order and Minimum Span: In practice we set the maximum possible lag order as $P_0 = 10$; i.e., $(p_{1,j}, p_{2,j}) \leq P_0$ for all j. Also, as mentioned before, we require each segment to have a minimum number of data points so that reasonable parameter estimates can be obtained. This requirement is called the minimum span constraint by [26]. For our problem, the minimum span $m_{p_{\max}}$ of a segment with a maximum lag order $p_{\max,j} = \max(p_{1,j}, p_{2,j})$ can be found in Table I.

3) Generating the First Generation Chromosomes: The way we generate the first-generation chromosomes is summarized in Algorithm 1. We denote a pre-specified parameter r_G as the probability for any time point t to be a breakpoint. See section B of the supplementary material for other methods for generating the first-generation chromosomes.

Once the first generation is available, we select parent chromosomes from it and apply the crossover and mutation operations to produce offspring chromosomes. We denote the pre-specified probability for performing a crossover operation as r_C, and the probability for mutation is $1 - r_C$.

4) Crossover: In the crossover operation, one offspring chromosome is generated in a manner that is summarized by Algorithm 2.

5) Mutation: During mutation, one offspring chromosome is produced by Algorithm 3.

Algorithm 2: Crossover.

Input: Chromosomes of last generation;
MDL values for last generation chromosomes;
Minimum span for different lag order: $m_{p_{\max}}$;
The upper bound of lag orders: P_0;
Output: New generation chromosome δ_{new}
initialize $\delta_{\text{new}} = (\delta_{\text{new},1}, \cdots, \delta_{\text{new,T}}) = (-1, \cdot, -1)$; $i = 1$;
Weight Sample 2 parent chromosomes $\delta_{p_1}, \delta_{p_2}$ based on their inverse MDL values;
Set $i = 1$.
while $i \leq T$ **do**
  Generate $r \sim Uniform(0, 1)$
  if $r \leq 0.5$ **then**
    SET $\delta_{new,i} = \delta_{p_1,i}$ **if** $\delta_{new,i} \neq -1$ **then**
      SET $p_{\max,i} = \max(\delta_{new,i}[0], \delta_{new,i}[1])$ $i = i + m_{p_{\max,i}} + 1$
    else
      $i = i + 1$
    end
  else
  end
end

6) Stopping Criterion: As mentioned in the previous section, the island model will be used, which allows migration after every M_i generation. The algorithm will finish if the chromosome with the smallest MDL value does not change for M_C consecutive migrations or if the total number of migrations exceeds an upper bound M_U. The chromosome with the smallest MDL value will be taken as the final solution provided by the algorithm.

Algorithm 4 provides an overall summary that links all the above ingredients of the genetic algorithm.

7) Refined Estimates for the Lag Orders: Although the above GA provides good results in estimating the number and locations of the breakpoints, the estimated locations are not always equal to the true ones. This could have negative impacts on the estimation of the lag orders if the estimated segment contains data points from its adjacent segments.

As Corollary II.1.1 states, although the estimated breakpoint locations are not necessarily to be exact, the true locations will likely be within some neighborhoods of the estimated ones. In light of this, when we estimate the lag orders $(p_{1,j}, p_{2,j})$ of the j-th estimated segment $[\hat{\tau}_{j-1}, \hat{\tau}_j)$, we only use data from $[\hat{\tau}_{j-1} + R_n, \hat{\tau}_j - R_n)$, where R_n is a calculated radius of the neighborhood, which can be calculated using the similar way as in [29]. The simulation results below show that this improves the estimation of the lag orders. We also only use those data from the shortened segment to estimate the regression parameters.

Fig. 1 shows the flowchart of this two-stage genetic algorithm.

IV. SIMULATION RESULTS

A. General Parameter Settings

We first specify the parameter values that we used in the GA in all simulation settings: upper bound of lag order $P_0 = 10$;

Authorized licensed use limited to: Univ of Calif Davis. Downloaded on August 23,2024 at 16:40:49 UTC from IEEE Xplore. Restrictions apply.

Algorithm 3: Mutation.

Input: Chromosomes of last generation;
MDL values for last generation chromosomes;
The probability of a gene mutating to non-breakpoint r_N;
The probability of a gene inherent from parent chromosome r_P;
Minimum span for different lag order: $m_{p_{\max}}$;
The upper bound of lag orders: P_0;

Output: New generation chromosome δ_{new}

initialize $\delta_{\text{new}} = (\delta_{\text{new,1}}, \cdots, \delta_{\text{new,T}}) = (-1, \cdot, -1)$; $i = 1$;
Weight Sample 1 parent chromosome δ_{p_1} based on their inverse MDL values;
Set $i = 1$.
while $i \leq T$ **do**
  Generate $r \sim Uniform(0, 1)$
  if i == *1* **then**
    if $r < r_P$ **then**
      SET $\delta_{new,i} = \delta_{p_1,i}$ **if** $\delta_{new,i} \neq -1$ **then**
        SET $p_{\max,i} = \max(\delta_{new,i}[0], \delta_{new,i}[1])$
        $i = i + m_{p_{\max,i}} + 1$
      else
        $i = i + 1$
      end
    else
      SAMPLE $p_{1,i}, p_{2,i} \sim [P_0]$ SET $\delta_i = (p_{1,i}, p_{2,i})$; $p_{\max,i} = \max(p_{1,i}, p_{2,i})$ $i = i + m_{p_{\max,i}} + 1$
    end
  else
    if $r < r_P$ **then**
      SET $\delta_{new,i} = \delta_{p_1,i}$ **if** $\delta_{new,i} \neq -1$ **then**
        SET $p_{\max,i} = \max(\delta_{new,i}[0], \delta_{new,i}[1])$
        $i = i + m_{p_{\max,i}} + 1$
      else
        $i = i + 1$
      end
    else if $r > r_G + r_N$ **then**
      SAMPLE $p_{1,i}, p_{2,i} \sim [P_0]$ SET $\delta_i = (p_{1,i}, p_{2,i})$; $p_{\max,i} = \max(p_{1,i}, p_{2,i})$ $i = i + m_{p_{\max,i}} + 1$
    $\delta_{new,i} = -1$ $i = i + 1$
  end
end

Algorithm 4: Genetic Algorithm for Solving (14).

Input: Minimum span for different lag order: $m_{p_{\max}}$;
Probability of Crossover: r_C;
Probability of being a breakpoint: r_G;
The probability of a gene mutating to non-breakpoint r_N;
The probability of a gene inherent from parent chromosome r_P;
The upper bound of lag orders: P_0;

Output: Final chromosome δ_{final}

Initialize chromosomes using Algorithm 1
while *minimum MDL value changes* **do**
  Generate $r \sim \text{Uniform}(0, 1)$
  if $r < r_C$ **then**
    Generate next generation using Crossover Algorithm 2
  else
    Generate next generation using Mutation Algorithm 3
  end
end

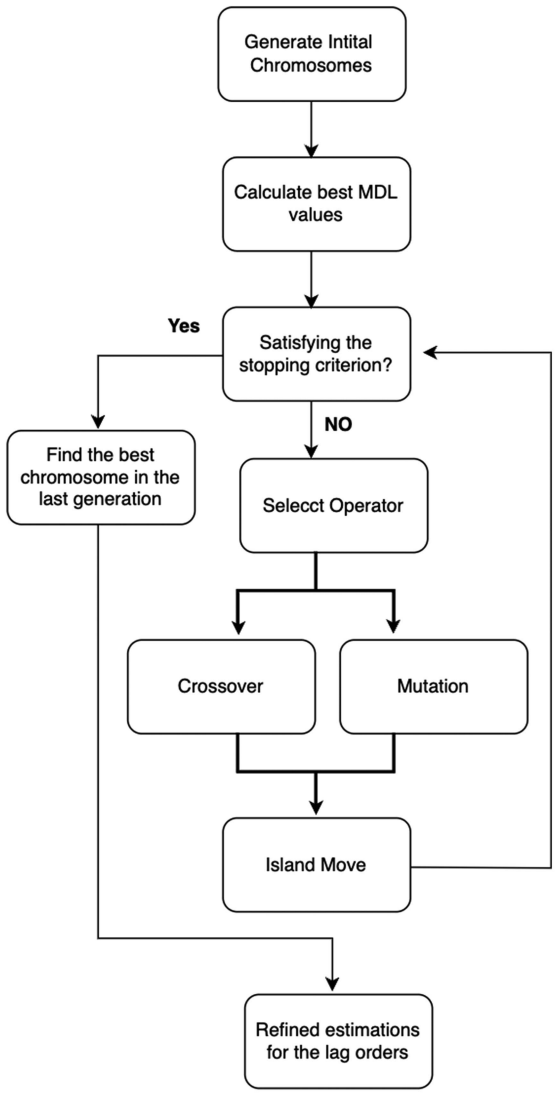

Fig. 1. Flowchart for the two-stage genetic algorithm.

number of islands $NI = 40$; number of chromosomes in each island $n_m = 40$; migration frequency $M_i = 5$; migration numbers $M_N = 2$; stopping criterion $M_C = 20$, $M_U = 100$; initialization probability $r_G = 0.1$; crossover probability $r_C = 0.9$; mutation probabilities $r_P = r_N = 0.3$; neighborhood radius $R_n = 0.5 \log(K) \log(T)$ as suggested in [29].

Three network structures from [3] are considered in each of the five simulation scenarios below:

1) *Power-Law Distribution Structure:* this structure mimics the phenomenon when a majority of nodes have very few edges while a few nodes have enormous numbers of edges.

Authorized licensed use limited to: Univ of Calif Davis. Downloaded on August 23,2024 at 16:40:49 UTC from IEEE Xplore. Restrictions apply.

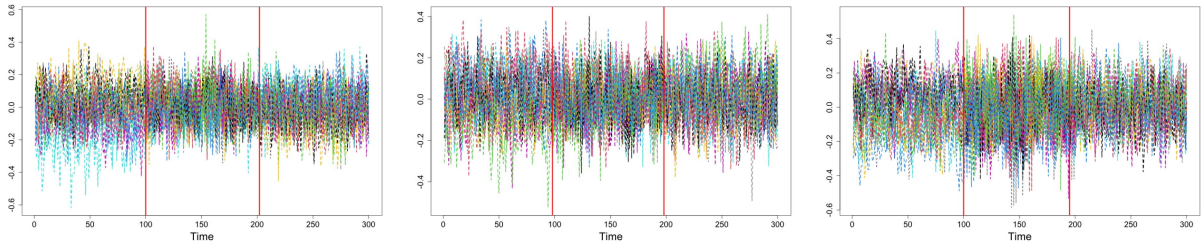

Fig. 2. Typical simulated data sets with estimated breakpoints (vertical red lines). Left: power-law network structure; middle: dyad independence network structure; right: stochastic block network structure.

A discrete power-law distribution is used to generate in-degree $d_i = \sum_{j \neq i} a_{ji}$, where $P(d_i = x) = cx^{-1.2}$ with a constant c set to $c = 1.5$. Then for each node i, randomly select d_i nodes to follow it.

2) *Dyad Independence Structure:* A dyad is defined as a pair of nodes $D_{ij} = (a_{ij,a_{ji}})$, $1 \leq i < j \leq K$ and it is assumed that different dyads are independent. We set $P(D_{ij} = (1,1)) = 0.1$, $P(D_{ij} = (1,0)) = P(D_{ij} = (0,1)) = 0.05$, and $P(D_{ij} = (0,0)) = 0.85$.
3) *Stochastic Block Structure:* Randomly assign each node a block label uniformly from 5 groups; i.e., $\{1, \ldots, 5\}$. We set $P(a_{ij} = 1) = 0.15$ if $\{i, j\}$ belong to the same group, and $P(a_{ij} = 1) = 0.015$ if $\{i, j\}$ belong to different groups. This implies that nodes within the same group will have higher chances of being connected.

The node specific covariates are generated as $\boldsymbol{V}_i = 0.15\boldsymbol{Z}_i$, where $\boldsymbol{Z}_i = (Z_{i1}, \ldots, Z_{i4})^T \in \mathbb{R}^4$ is from a multivariate normal distribution with $\mathcal{N}_4(\boldsymbol{0}, \boldsymbol{\Sigma}_Z)$ with $\boldsymbol{\Sigma}_Z = (\sigma_{j_1 j_2}) = (0.5^{|j_1 - j_2|})$w. For all different scenarios below, we set the number of time points $T = 300$, the number of nodes $K = 20$, and the number of simulation runs to 100.

B. Scenario 1: Impact of the Refining Step of Section III-B7

In this first scenario, breakpoints are set as $\mathcal{T} = (\frac{T}{3}, \frac{2T}{3}) = (100, 200)$, with two sets of error variances: $\sigma_1 = \sigma_2 = \sigma_3 = 0.1$ and $\sigma_1 = \sigma_2 = \sigma_3 = 0.3$. We have $p_{1,j} = p_{2,j}$ for all segments:

Segment 1: $p_{1,1} = p_{2,1} = 1$, $\boldsymbol{\theta}_1 = (\beta_{0,1}, \alpha_{1,1}, \beta_{1,1}, \boldsymbol{\gamma}_1) = (0, -0.1, 0.2, 0.1, 0.4, 0.1, 0.2) \in \mathbb{R}^7$.

Segment 2: $p_{1,2} = p_{2,2} = 2$, $\boldsymbol{\theta}_2 = (\beta_{0,2}, \alpha_{1,2}, \alpha_{2,2}, \beta_{1,2}, \beta_{2,2}, \boldsymbol{\gamma}_2) = (0, 0.2, -0.22, -0.12, 0.4, -0.1, 0.1, 0.2, -0.1) \in \mathbb{R}^9$.

Segment 3: $p_{1,3} = p_{2,3} = 2$, $\boldsymbol{\theta}_3 = (\beta_{0,3}, \alpha_{1,3}, \alpha_{2,3}, \beta_{1,3}, \beta_{2,3}, \boldsymbol{\gamma}_3) = (0, -0.12, 0.1, 0.25, -0.4, 0.2, -0.5, 0.1, 0.1) \in \mathbb{R}^9$.

The above regression coefficients guarantee that the NAR model in each segment is stationary [3], [6].

For each of the three network structures, 100 data sets were generated, and the proposed method was applied to estimate the breakpoints and other model parameters. One typical data set (with $\sigma_1 = \sigma_2 = \sigma_3 = 0.1$) from each of the network structures are shown in Fig. 2, while the breakpoint estimation results for the 100 runs are summarized in Tables II and III respectively for smaller and larger error variances. One can observe that the proposed method successfully detected the correct number of breakpoints in all cases when error variances were smaller. The method also produced excellent estimates for the breakpoint locations, as shown by their mean values and standard deviations.

TABLE II
RESULTS OF SELECTED BREAKPOINTS OF SCENARIO 1 WHEN $\sigma_1 = \sigma_2 = \sigma_3 = 0.1$

	Breakpoint	Truth	Mean	SD	Selection Rate
Power-Law Structure	1	0.333	0.323	0.016	100%
	2	0.667	0.666	0.008	100%
Dyad Independence Structure	1	0.333	0.325	0.016	100%
	2	0.667	0.665	0.009	100%
Stochastic Block Structure	1	0.333	0.328	0.010	100%
	2	0.667	0.665	0.011	100%

Truth: locations of the true breakpoints; Mean/SD: means/standard deviations of the estimated breakpoint locations over the 100 simulation runs; Selection Rate: percentages of times that the correct number of breakpoints were selected.

TABLE III
SIMILAR TO TABLE II BUT FOR $\sigma_1 = \sigma_2 = \sigma_3 = 0.3$

	Breakpoint	Truth	Mean	SD	Selection Rate
Power-Law Structure	1	0.333	0.320	0.020	93%
	2	0.667	0.669	0.015	91%
Dyad Independence Structure	1	0.333	0.321	0.015	90%
	2	0.667	0.663	0.009	93%
Stochastic Block Structure	1	0.333	0.330	0.011	94%
	2	0.667	0.664	0.012	93%

We compare the estimation results of the lag orders without and with the refining step described in Section III-B7: the results for the smaller error variance cases are summarized, respectively, in Tables IV and V. The results for the larger error variance cases are similar and hence omitted for brevity. One can observe that the refining step did indeed improve the estimation results of the lag orders. So if not specified, the refining step was applied in all the numerical work presented below.

The estimation results of the regression parameters $(\theta_1, \theta_2, \theta_3)$ are delayed to section C in the supplementary material.

C. Scenario 2: Different Segment Lengths

In the second scenario, two sets of breakpoints were used: $\mathcal{T} = (\frac{T}{3}, \frac{2T}{3}) = (100, 200)$ and $\mathcal{T} = (\frac{T}{6}, \frac{5T}{6}) = (50, 250)$. The error variances are $\sigma_1 = \sigma_2 = \sigma_3 = 0.1$, and other model parameters are:

Segment 1: $p_{1,1} = p_{2,1} = 1$, $\boldsymbol{\theta}_1 = (\beta_{0,1}, \alpha_{1,1}, \beta_{1,1}, \boldsymbol{\gamma}_1) = (0, -0.1, 0.2, 0.1, 0.4, 0.1, 0.2) \in \mathbb{R}^7$.

Authorized licensed use limited to: Univ of Calif Davis. Downloaded on August 23,2024 at 16:40:49 UTC from IEEE Xplore. Restrictions apply.

TABLE IV
ESTIMATED LAG ORDERS IN EACH SEGMENT OF THREE NETWORK STRUCTURES OF SCENARIO 1 WITHOUT THE REFINING STEP OF SECTION III-B7, WHERE $\sigma_1 = \sigma_2 = \sigma_3 = 0.1$

	Orders	0	1	2	3	4	5
Power-Law Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0.05	**0.88**	0.06	0.01	0
	$p_{2,2}$	0	0.05	**0.85**	0.09	0.01	0
	$p_{1,3}$	0	0.12	**0.78**	0.08	0.02	0
	$p_{2,3}$	0	0.10	**0.78**	0.10	0.02	0
Dyad Independence Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0.11	**0.83**	0.05	0.01	0
	$p_{2,2}$	0	0.10	**0.80**	0.06	0.04	0
	$p_{1,3}$	0	0.23	**0.73**	0.04	0	0
	$p_{2,3}$	0	0.20	**0.75**	0.01	0.03	0.01
Stochastic Block Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0	**0.80**	0.20	0	0
	$p_{2,2}$	0	0	**0.83**	0.17	0	0
	$p_{1,3}$	0	0.12	**0.65**	0.19	0.04	0
	$p_{2,3}$	0	0.10	**0.70**	0.20	0	0

The numbers are the proportions that a particular order was estimated. Bolded numbers correspond to the true orders.

TABLE V
SIMILAR TO TABLE IV BUT WITH THE REFINING STEP OF SECTION III-B7

	Orders	0	1	2	3	4	5
Power-Law Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0	**1**	0	0	0
	$p_{2,2}$	0	0.01	**0.99**	0	0	0
	$p_{1,3}$	0	0.02	**0.98**	0	0	0
	$p_{2,3}$	0	0.02	**0.98**	0	0	0
Dyad Independence Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0	**1**	0	0	0
	$p_{2,2}$	0	0	**1**	0	0	0
	$p_{1,3}$	0	0	**0.97**	0.03	0	0
	$p_{2,3}$	0	0.02	**0.96**	0.02	0	0
Stochastic Block Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0	**1**	0	0	0
	$p_{2,2}$	0	0.01	**0.99**	0	0	0
	$p_{1,3}$	0	0	**1**	0	0	0
	$p_{2,3}$	0	0.01	**0.98**	0.01	0	0

The numbers are the proportions that a particular order was estimated. Bolded numbers correspond to the true orders.

TABLE VI
SIMILAR TO TABLE II BUT FOR SCENARIO 2 WITH TRUE BREAKPOINTS $\mathcal{T} = (100, 200)$

	Breakpoint	Truth	Mean	SD	Selection Rate
Power-Law Structure	1	0.333	0.330	0.004	100%
	2	0.667	0.662	0.010	100%
Dyad Independence Structure	1	0.333	0.330	0.009	100%
	2	0.667	0.660	0.011	100%
Stochastic Block Structure	1	0.333	0.332	0.007	100%
	2	0.667	0.662	0.010	100%

Segment 2: $p_{1,2} = 2, p_{2,2} = 1$, $\boldsymbol{\theta}_2 = (\beta_{0,2}, \alpha_{1,2}, \alpha_{2,2}, \beta_{1,2}, \boldsymbol{\gamma}_2) = (0, 0.2, -0.22, -0.12, -0.1, 0.1, 0.2, -0.1) \in \mathbb{R}^8$.

Segment 3: $p_{1,3} = 1, p_{2,3} = 2$, $\boldsymbol{\theta}_3 = (\beta_{0,3}, \alpha_{1,3}, \beta_{1,3}, \beta_{2,3}, \boldsymbol{\gamma}_3) = (0, -0.12, 0.25, -0.4, 0.2, -0.5, 0.1, 0.1) \in \mathbb{R}^8$.

The estimated breakpoints from 100 simulation runs are summarized in Tables VI and VII , in a similar fashion as in Scenario 1. The proposed method selected the correct number of breakpoints for all cases. It also gave excellent location estimates, as reflected by the mean and standard deviations.

TABLE VII
SIMILAR TO TABLE II BUT FOR SCENARIO 2 WITH TRUE BREAKPOINTS $\mathcal{T} = (50, 250)$

	Breakpoint	Truth	Mean	SD	Selection Rate
Power-Law Structure	1	0.167	0.167	0.012	100%
	2	0.833	0.830	0.007	100%
Dyad Independence Structure	1	0.167	0.166	0.011	100%
	2	0.833	0.829	0.010	100%
Stochastic Block Structure	1	0.167	0.165	0.007	100%
	2	0.833	0.831	0.006	100%

TABLE VIII
SIMILAR TO TABLE V BUT FOR SCENARIO 2 WITH TRUE BREAKPOINTS $\mathcal{T} = (100, 200)$

	Orders	0	1	2	3	4	5
Power-Law Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0.01	**0.99**	0	0	0
	$p_{2,2}$	0	**0.99**	0.01	0	0	0
	$p_{1,3}$	0	**0.96**	0.04	0	0	0
	$p_{2,3}$	0	0.01	**0.99**	0	0	0
Dyad Independence Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0.05	**0.95**	0	0	0
	$p_{2,2}$	0	**1**	0	0	0	0
	$p_{1,3}$	0	**0.98**	0.02	0	0	0
	$p_{2,3}$	0	0.02	**0.98**	0	0	0
Stochastic Block Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0.02	**0.98**	0	0	0
	$p_{2,2}$	0	**0.96**	0.04	0	0	0
	$p_{1,3}$	0	**0.98**	0.02	0	0	0
	$p_{2,3}$	0	0	**1**	0	0	0

The numbers are the proportions that a particular order was estimated. Bolded numbers correspond to the true orders.

TABLE IX
SIMILAR TO TABLE V BUT FOR SCENARIO 2 WITH TRUE BREAKPOINTS $\mathcal{T} = (50, 250)$

	Orders	0	1	2	3	4	5
Power-Law Structure	$p_{1,1}$	0	**0.98**	0.02	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0.04	**0.96**	0	0	0
	$p_{2,2}$	0	**0.99**	0.01	0	0	0
	$p_{1,3}$	0	**0.89**	0.11	0	0	0
	$p_{2,3}$	0	0.05	**0.95**	0	0	0
Dyad Independence Structure	$p_{1,1}$	0	**0.99**	0.01	0	0	0
	$p_{2,1}$	0	**0.99**	0.01	0	0	0
	$p_{1,2}$	0	0.11	**0.89**	0	0	0
	$p_{2,2}$	0	**0.96**	0.04	0	0	0
	$p_{1,3}$	0	**0.92**	0.08	0	0	0
	$p_{2,3}$	0	0.01	**0.97**	0.02	0	0
Stochastic Block Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0.04	**0.96**	0	0	0
	$p_{2,2}$	0	**0.94**	0.06	0	0	0
	$p_{1,3}$	0	**0.90**	0.10	0	0	0
	$p_{2,3}$	0	0.05	**0.95**	0	0	0

The numbers are the proportions that a particular order was estimated. Bolded numbers correspond to the true orders.

Estimation results of the lag orders are summarized in Tables VIII and IX. The results seem to be slightly worse for breakpoints $\mathcal{T} = (50, 250)$ when compared to $\mathcal{T} = (100, 200)$. One possible explanation is that the first and third segments are

Authorized licensed use limited to: Univ of Calif Davis. Downloaded on August 23,2024 at 16:40:49 UTC from IEEE Xplore. Restrictions apply.

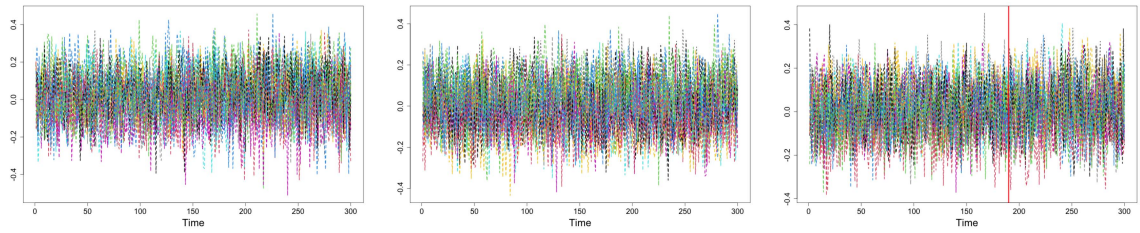

Fig. 3. Typical simulated data sets with a slowly varying coefficient. Left: power-law network structure with no detected breakpoint; middle: dyad independence network structure with no detected breakpoint; right: stochastic block network structure with one detected breakpoint (red vertical line).

TABLE X
SIMILAR TO TABLE II BUT FOR SCENARIO 3

	Breakpoint	Truth	Mean	SD	Selection Rate
Power-Law Structure	1	0.333	0.324	0.011	100%
	2	0.667	0.663	0.007	100%
Dyad Independence Structure	1	0.333	0.325	0.010	100%
	2	0.667	0.664	0.01	100%
Stochastic Block Structure	1	0.333	0.323	0.012	100%
	2	0.667	0.664	0.007	100%

TABLE XI
SIMILAR TO TABLE V BUT FOR SCENARIO 3

	Orders	0	1	2	3	4	5
Power-Law Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0	**0.99**	0.01	0	0
	$p_{2,2}$	0	**1**	0	0	0	0
	$p_{1,3}$	0	**0.97**	0.03	0	0	0
	$p_{2,3}$	0	0.02	**0.98**	0	0	0
Dyad Independence Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0.03	**0.97**	0	0	0
	$p_{2,2}$	0	**0.99**	0.01	0	0	0
	$p_{1,3}$	0	**0.98**	0.02	0	0	0
	$p_{2,3}$	0	0.02	**0.98**	0	0	0
Stochastic Block Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0.02	**0.98**	0	0	0
	$p_{2,2}$	0	**0.95**	0.05	0	0	0
	$p_{1,3}$	0	**0.99**	0.01	0	0	0
	$p_{2,3}$	0	0	**1**	0	0	0

The numbers are the proportions that a particular order was estimated. Bolded numbers correspond to the true orders.

shorter, and hence there were less number of data points available for the estimation of the lag orders.

D. Scenario 3: Correlated Variance Matrices

Here we set the breakpoints as $\mathcal{T} = (\frac{T}{3}, \frac{2T}{3}) = (100, 200)$ and adopt a more complicated error structure: the errors are correlated and the covariance matrix of the error terms is dense. To be more specific, we assume the covariance matrix of error $\boldsymbol{\varepsilon}_j$ is $\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}_j} = 0.01((\sigma_{ij}))_{T\times T}$ with $\sigma_{ij} = 0.5^{|i-j|}$. All the remaining model parameters are the same as those in Scenario 2.

The estimation results for breakpoints and lag orders are summarized, respectively, in Tables X and XI. One can observe that the proposed method performed very well in this scenario, which confirms the applicability of the method in the case of correlated error terms. These results verify our previous claim that the variances of the error terms do not have to be the same for the proposed method to work.

TABLE XII
SIMILAR TO TABLE II BUT FOR SCENARIO 5 WITH $\pi_T \sim N(0, 0.1^2)$

	Breakpoint	Truth	Mean	SD	Selection Rate
Power-Law Structure	1	0.333	0.330	0.009	100%
	2	0.667	0.660	0.008	100%
Dyad Independence Structure	1	0.333	0.320	0.007	100%
	2	0.667	0.664	0.009	100%
Stochastic Block Structure	1	0.333	0.330	0.013	100%
	2	0.667	0.663	0.010	100%

TABLE XIII
SIMILAR TO TABLE II BUT FOR SCENARIO 5 WITH $\pi_T \sim N(0, 0.5^2)$

	Breakpoint	Truth	Mean	SD	Selection Rate
Power-Law Structure	1	0.333	0.331	0.010	94%
	2	0.667	0.662	0.012	92%
Dyad Independence Structure	1	0.333	0.323	0.010	95%
	2	0.667	0.662	0.008	92%
Stochastic Block Structure	1	0.333	0.331	0.015	91%
	2	0.667	0.661	0.012	93%

E. Scenario 4: Slowly Varying Coefficient

In this scenario we consider the case that there is no breakpoint and one of the coefficients is slowly varying. The exact specification is: $p_1 = p_2 = 1$ and $\boldsymbol{\theta} = (\beta_0, \alpha_1, \beta_1, \boldsymbol{\gamma}_1) = (0, a_t, -0.1, 0.1, 0.4, 0.1, 0.2)$, where $a_t = 0.55 - 0.25\cos(\pi t/T)$ changes over time. Typical realizations of this process are shown in Fig. 3 with breakpoints estimated by the proposed method.

For both the Power-Law and Dyad Independence Structures, no breakpoint was detected in any of the simulation runs, while for the Stochastic Block Structure, one breakpoint was always detected near $0.6T$. The reason may be that the Stochastic Block structure has the lowest structure sparsity, so it contains less information in this difficult scenario.

F. Scenario 5: Mis-Specified $\boldsymbol{W}$

In this last scenario, the row normalized network matrix $\boldsymbol{W}$ is mis-specified. Although $\boldsymbol{W}$ is always assumed known and can be derived directly from the network structure matrix $\boldsymbol{A}$ in NAR models, in many applications, it is reasonable to assume that it is empirically defined and hence it may not be exactly accurate [6]. Here we use $\boldsymbol{W}_{obs} = \boldsymbol{W} + \pi_T$ with $\pi_T \sim N(0, 0.1^2)$

Authorized licensed use limited to: Univ of Calif Davis. Downloaded on August 23,2024 at 16:40:49 UTC from IEEE Xplore. Restrictions apply.

TABLE XIV
SIMILAR TO TABLE V BUT FOR SCENARIO 5 WITH $\pi_T \sim N(0, 0.1^2)$

	Orders	0	1	2	3	4	5
Power-Law Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0.02	**0.98**	0	0	0
	$p_{2,2}$	0	**0.97**	0.03	0	0	0
	$p_{1,3}$	0	**0.95**	0.05	0	0	0
	$p_{2,3}$	0	0.02	**0.98**	0	0	0
Dyad Independence Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0.04	**0.96**	0	0	0
	$p_{2,2}$	0	**0.99**	0.01	0	0	0
	$p_{1,3}$	0	**0.98**	0.02	0	0	0
	$p_{2,3}$	0	0.02	**0.98**	0	0	0
Stochastic Block Structure	$p_{1,1}$	0	**1**	0	0	0	0
	$p_{2,1}$	0	**1**	0	0	0	0
	$p_{1,2}$	0	0.02	**0.98**	0	0	0
	$p_{2,2}$	0	**0.97**	0.03	0	0	0
	$p_{1,3}$	0	**1**	0	0	0	0
	$p_{2,3}$	0	0.01	**0.99**	0	0	0

The numbers are the proportions that a particular order was estimated. Bolded numbers correspond to the true orders.

TABLE XV
SIMILAR TO TABLE V BUT FOR SCENARIO 5 WITH $\pi_T \sim N(0, 0.5^2)$

	Orders	0	1	2	3	4	5
Power-Law Structure	$p_{1,1}$	0	**0.98**	0	0	0	0.02
	$p_{2,1}$	0.01	**0.89**	0	0	0	0.1
	$p_{1,2}$	0	0.05	**0.90**	0	0	0.05
	$p_{2,2}$	0	**0.91**	0.06	0	0	0.03
	$p_{1,3}$	0	**0.90**	0.06	0	0	0.04
	$p_{2,3}$	0	0.04	**0.92**	0	0	0.04
Dyad Independence Structure	$p_{1,1}$	0	**0.95**	0.02	0.03	0	0
	$p_{2,1}$	0	**0.96**	0	0	0	0.04
	$p_{1,2}$	0	0.06	**0.89**	0	0	0.05
	$p_{2,2}$	0.08	**0.87**	0.01	0	0	0.05
	$p_{1,3}$	0	**0.88**	0.07	0	0	0.05
	$p_{2,3}$	0	0.02	**0.89**	0	0	0.09
Stochastic Block Structure	$p_{1,1}$	0	**0.97**	0.03	0	0	0
	$p_{2,1}$	0	**0.99**	0.01	0	0	0
	$p_{1,2}$	0	0.07	**0.90**	0	0	0.03
	$p_{2,2}$	0	**0.88**	0.05	0	0	0.07
	$p_{1,3}$	0	**0.92**	0	0	0	0.08
	$p_{2,3}$	0	0.02	**0.89**	0	0	0.09

The numbers are the proportions that a particular order was estimated. Bolded numbers correspond to the true orders.

and $\pi_T \sim N(0, 0.5^2)$. Other model parameters are the same as those in Scenario 2 with true breakpoints $\mathcal{T} = (100, 200)$.

The estimation results for the breakpoints and lag orders are summarized in Tables XIIto XV. These results are comparable to those from Scenario 2, which suggests that the introduction of the error term π_T did not affect the estimation results too much. In other words, the results suggest that the proposed method is, to a certain extent, robust against changes in the network structure matrix.

V. REAL DATA ANALYSIS

This section applies the proposed method to a Manhattan yellow cab demand data set, which was obtained from the NYC Taxi and Limousine Commission's website.[1] This dataset depicts the number of yellow cab pick-ups in different taxi zones and is aggregated spatially over the zipcodes. Here, Manhattan was divided into 64 taxi zones, as shown in Fig. 4. Note that

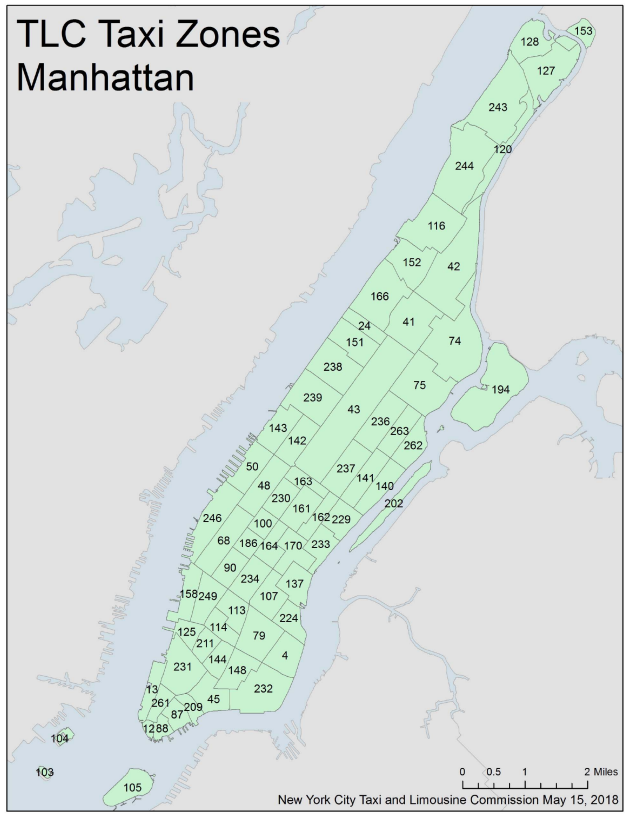

Fig. 4. Taxi Zones of Manhattan[2].

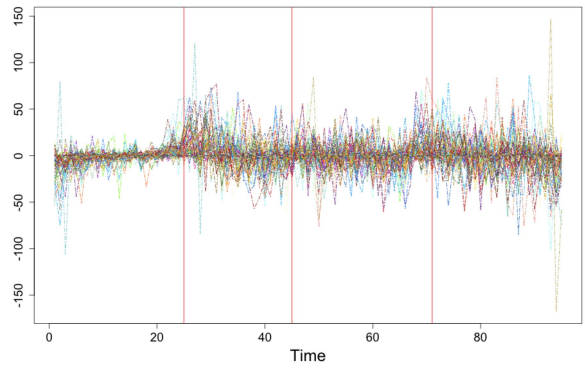

Fig. 5. The one-lag difference time series of the Manhattan yellow cab data set in 59 taxi zones over 96 time points taken on April 16th, 2014. Altogether 3 breakpoints were detected (red vertical lines).

zones 103, 104, 105, 202, and 194 are isolated with no common boundaries with any other zones, thus these five zones were not included in the analysis.

For the remaining 59 zones, we aggregated the numbers of their yellow cab pick-ups temporally over 15-minutes intervals for the date April 16th, 2014. Thus, there are $T = 96$ time points with $K = 59$ nodes at each time point. The network structure $\boldsymbol{A} = (a_{i_1,i_2}) \in \mathbb{R}^{59\times 59}$ was constructed by using the physical relationships of these taxi zones: $a_{i_1,i_2} = 1$ if zones i_1 and i_2 share a common boundary, otherwise $a_{i_1,i_2} = 0$. We considered

[1]Online. [Available]: https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

[2]Online. [Available]: https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

Authorized licensed use limited to: Univ of Calif Davis. Downloaded on August 23,2024 at 16:40:49 UTC from IEEE Xplore. Restrictions apply.

the average tip amount of the trips in different zones in April 2014 as the node-specified exogenous covariates.

We performed a first-order difference to the data to remove the first-order non-stationarities. Altogether, three breakpoints were detected by the proposed method; they are shown in Fig. 5. These three breakpoints correspond to 6:15 AM, 11:30 AM, and 5:45 PM, which seem to coincide with the daily major changes in traffic patterns in Manhattan: people commute to work, go out for lunch, and return home after work. These results broadly agree with those in [29]. Lastly, the fitting time for this data set is about 46.7 seconds per generation on a 2020 Macbook Pro with an M1 chip.

VI. Concluding Remarks

This paper developed a method for simultaneous multiple breakpoint detection and parameter estimation for piecewise stationary NAR models. The proposed method utilizes the MDL principle to derive an objective criterion for estimating the breakpoints as well as other model parameters. It has been shown that the MDL estimates enjoy desirable asymptotic properties. To optimize the MDL objective criterion, the proposed method uses a tailor-made GA. Through a sequence of simulation experiments, the proposed method is shown to possess excellent empirical properties. Lastly, the proposed method was applied to analyze a Manhattan yellow cab data set and yielded similar results as those reported in the literature.

The proposed methodology offers several notable advantages, such as its statistical consistency and the straightforward interpretation offered by piecewise stationary NAR models. However, it is not without its limitations. One key constraint is its reliance on the piecewise stationary assumption; deviation from this assumption could lead to the identification of breakpoints that do not truly exist, such as in cases where the data undergo gradual changes without clear breakpoints. Additionally, the method presupposes an observed network structure that stays constant throughout the observation period, a condition that may not always hold. While network imputation techniques [30], [31], [32] offer a remedy by enabling the reconstruction of the missing network structure, they introduce a layer of uncertainty to the analysis.

Future work includes two ambitious goals. First, we aim to enhance the method by incorporating uncertainty quantification for the fitted piecewise stationary NAR model. This development would provide a deeper understanding of the model's predictive confidence across different segments. Second, we plan to explore strategies for condensing the network without losing critical information. This endeavor will investigate approaches similar to factor modeling in high-dimensional time series analysis, aiming to maintain the important information hidden in the data while simplifying the model's complexity.

References

[1] J. Y. Campbell, G. Chacko, J. Rodriguez, and L. M. Viceira, "Strategic asset allocation in a continuous-time VAR model," *J. Econ. Dyn. Control*, vol. 28, no. 11, pp. 2195–2214, 2004.

[2] A. Cologni and M. Manera, "Oil prices, inflation and interest rates in a structural cointegrated VAR model for the G-7 countries," *Energy Econ.*, vol. 30, no. 3, pp. 856–888, 2008.

[3] X. Zhu, R. Pan, G. Li, Y. Liu, and H. Wang, "Network vector autoregression," *Ann. Statist.*, vol. 45, no. 3, pp. 1096–1123, 2017.

[4] D. Katselis, C. L. Beck, and R. Srikant, "Mixing times and structural inference for Bernoulli autoregressive processes," *IEEE Trans. Netw. Sci. Eng.*, vol. 6, no. 3, pp. 364–378, Jul.–Sep. 2019.

[5] K. Yang, S. Dou, P. Luo, X. Wang, and H. V. Poor, "Robust group anomaly detection for quasi-periodic network time series," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 4, pp. 2833–2845, Jul./Aug. 2022.

[6] H. Yin, A. Safikhani, and G. Michailidis, "A general modeling framework for network autoregressive processes," *Technometrics,* vol. 65, pp. 579–589, 2023.

[7] M. Knight, K. Leeming, G. Nason, and M. Nunes, "Generalised network autoregressive processes and the GNAR package," *J. Statist. Softw.*, vol. 96, pp. 1–36, 2019.

[8] H. Ombao, R. Von Sachs, and W. Guo, "SLEX analysis of multivariate nonstationary time series," *J. Amer. Stat. Assoc.*, vol. 100, no. 470, pp. 519–531, 2005.

[9] G. E. Primiceri, "Time varying structural vector autoregressions and monetary policy," *Rev. Econ. Stud.*, vol. 72, no. 3, pp. 821–852, 2005.

[10] J. Rissanen, *Stochastic Complexity in Statistical Inquiry*. Singapore: World Scientific, 1989.

[11] J. Rissanen, *Information and Complexity in Statistical Modeling*. Berlin, Germany: Springer, 2007.

[12] L. Chen, J. Zhou, and L. Lin, "Hypothesis testing for populations of networks," *Commun. Statist.-Theory Methods*, vol. 52, pp. 3661–3684, 2023.

[13] S. Banerjee and K. Guhathakurta, "Change-point analysis in financial networks," *Stat*, vol. 9, no. 1, 2020, Art. no. e269.

[14] J. Flossdorf and C. Jentsch, "Change detection in dynamic networks using network characteristics," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 7, pp. 451–464, 2021.

[15] T. Zhu, P. Li, L. Yu, K. Chen, and Y. Chen, "Change point detection in dynamic networks based on community identification," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 3, pp. 2067–2077, Jul.–Sep. 2020.

[16] L. Peel and A. Clauset, "Detecting change points in the large-scale structure of evolving networks," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2914–2920.

[17] S. Moreno and J. Neville, "Network hypothesis testing using mixed Kronecker product graph models," in *Proc. IEEE 13th Int. Conf. Data Mining*, 2013, pp. 1163–1168.

[18] Y. Hulovatyy and T. Milenković, "SCOUT: Simultaneous time segmentation and community detection in dynamic networks," *Sci. Rep.*, vol. 6, no. 1, pp. 1–11, 2016.

[19] T. C. M. Lee, "Segmenting images corrupted by correlated noise," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 5, pp. 481–492, May 1998.

[20] R. R. Bouckaert, "Probabilistic network construction using the minimum description length principle," in *Proc. Eur. Conf. Symbolic Quantitative Approaches Reasoning Uncertainty*, 1993, pp. 41–48.

[21] R. C. Cheung, A. Aue, S. Hwang, and T. C. M. Lee, "Simultaneous detection of multiple change points and community structures in time series of networks," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 6, pp. 580–591, 2020.

[22] C. Xu and T. C. M. Lee, "Statistical consistency for change point detection and community estimation in time-evolving dynamic networks," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 8, pp. 215–227, 2022.

[23] A. Aue, R. C. Y. Cheung, T. C. M. Lee, and M. Zhong, "Segmented model selection in quantile regression using the minimum description length principle," *J. Amer. Stat. Assoc.*, vol. 109, pp. 1241–1256, 2014.

[24] T. C. M. Lee, "Regression spline smoothing using the minimum description length principle," *Statist. Probability Lett.*, vol. 48, pp. 71–82, 2000.

[25] T. C. M. Lee, "An introduction to coding theory and the two–part minimum description length principle," *Int. Stat. Rev.*, vol. 69, pp. 169–183, 2001.

[26] R. A. Davis, T. C. M. Lee, and G. A. Rodriguez-Yam, "Structural break estimation for nonstationary time series models," *J. Amer. Stat. Assoc.*, vol. 101, no. 473, pp. 223–239, 2006.

[27] A. S. Fraser, "Simulation of genetic systems by automatic digital computers I. Introduction," *Australian J. Biol. Sci.*, vol. 10, no. 4, pp. 484–491, 1957.

[28] V. S. Gordan and D. Whitley, "Serial and parallel genetic algorithms as function optimizers," in *Proc. 5th Int. Conf. Genet. Algorithms*, 1993, pp. 177–183.

Authorized licensed use limited to: Univ of Calif Davis. Downloaded on August 23,2024 at 16:40:49 UTC from IEEE Xplore. Restrictions apply.

[29] A. Safikhani and A. Shojaie, "Joint structural break detection and parameter estimation in high-dimensional nonstationary VAR models," *J. Amer. Stat. Assoc.*, vol. 117, no. 537, pp. 251–264, 2022.
[30] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Sci. Rep.*, vol. 8, no. 1, 2018, Art. no. 6085.
[31] M. Sangeetha and M. Senthil Kumaran, "Deep learning-based data imputation on time-variant data using recurrent neural network," *Soft Comput.*, vol. 24, pp. 13369–13380, 2020.
[32] J. Yoon, J. Jordon, and M. Schaar, "GAIN: Missing data imputation using generative adversarial nets," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5689–5698.

Yi Han received the B.A. degree in statistics from Wuhan University, Wuhan, China, in 2019. He is currently working toward the Ph.D. degree in statistics with the University of California, Davis, Davis, CA, USA. His research interests include change point detection, network analysis, and statistical applications in other disciplines.

Thomas C. M. Lee (Senior Member, IEEE) received the B.App.Sc. (Math) degree and the B.Sc. (Hons.) (Math) degree with University Medal from the University of Technology Sydney, Sydney, NSW, Australia, in 1992 and 1993, respectively, and the Ph.D. degree jointly from Macquarie University, Sydney, NSW, and CSIRO Mathematical and Information Sciences, Sydney, NSW, in 1997. From 2015 to 2018, he was the Chair of the Department of Statistics, University of California, Davis, Davis, CA, USA. He is currently a Professor of statistics and Associate Dean of the Faculty for Mathematical and Physical Sciences, University of California, Davis. His research interests include inference methods, image and signal processing, and statistical applications in other scientific disciplines. He is an elected Fellow of the American Association for the Advancement of Science (AAAS), American Statistical Association (ASA), and Institute of Mathematical Statistics (IMS). From 2013 to 2015, he was the Editor-in-Chief of the *Journal of Computational and Graphical Statistics*.

Authorized licensed use limited to: Univ of Calif Davis. Downloaded on August 23,2024 at 16:40:49 UTC from IEEE Xplore. Restrictions apply.