Automatic Time-Series Clustering via Network Inference

Kohei Obata[1], Yasuko Matsubara[1], Koki Kawabata[1] and Yasushi Sakurai[1]

[1]*SANKEN, Osaka University*

Abstract

Given a collection of multidimensional time-series that contains an unknown type and number of network structures between variables, how efficiently can we find typical patterns and their points of variation? How can we interpret important relationships with obtained patterns? In this paper, we propose a new method of model-based clustering, which we call network clustering via graphical lasso (NGL). Our method has the following properties: (a) *Interpretable*: it provides interpretable network structures and cluster assignments for the data; (b) *Automatic*: it determines the optimal cut points and the number of clusters automatically; (c) *Accurate*: it provides reliable clustering performance thanks to the automated algorithm. We evaluate our NGL algorithm on both real and synthetic datasets, obtaining interpretable network structure results and outperforming state-of-the-art baselines in terms of accuracy.

Keywords

time-series, network structure, graphical lasso,

1. Introduction

Many applications generate time-series data including those used in automobiles [1], biology, social networks and in relation to financial data. In most cases, these data are multidimensional, and it is important to find typical patterns, which have a specific network structure. In practice, real-life data have multiple distinct patterns, which differentiate their network structures. For example, automobile sensor data from a driving session can be composed of some basic actions and some abrupt actions (i.e., going straight, turning right, turning left, slowing down, sudden braking, sudden turning). The network structure is equal to the graph structure. In this case, sensors can be represented as nodes, and sensor interactions can be represented as edges. For a turning action, lateral acceleration and steering angle may have an edge and for a braking action, brake pedal stroke and longitudinal acceleration may have an edge.

In this paper, we focus on finding a network structure automatically from multidimensional time-series data. Understanding the structure of these networks is useful because it allows us to devise models of sensor interaction, which can be used to analyze such behaviours as fossil-efficient driving. However, there are many network structures in the data, which change over time, and it is difficult to find a meaningful segmentation point since no one knows how data change. Moreover, the number of clusters should be selected automatically to find abrupt changes or for an extension to online learning, because in most cases, we do not know the optimal number of clusters in advance.

VLDB'22, Sep 05–9, 2022, Sydney, Australia

obata88@sanken.oska-u.ac.jp (K. Obata);
yasuko@sanken.osaka-u.ac.jp (Y. Matsubara);
koki@sanken.osaka-u.ac.jp (K. Kawabata);
yasushi@sanken.osaka-u.ac.jp (Y. Sakurai)

In this paper, we propose an automatic algorithm, called network clustering via graphical lasso (NGL), which enables us to summarize multidimensional time-series into meaningful patterns efficiently based on the graphical lasso problem. Intuitively, the problem we wish to solve is as follows.

InformalProblem 1. ***Given*** *a large collection of multidimensional time-series data with underlying network structures X,* ***Find*** *a compact description of X, which consists of:*

1. *a set of segments and their cut points*
2. *a set of segment groups (i.e., clusters) of similar network structures*
3. *the optimal number of clusters*

Contrast with Competitors. We will compare NGL with existing methods from the viewpoint of network inference. Network estimation with time-series information has been studied as a method for analyzing economic data and biological signal data because of the high interpretability of its graphical model [2]. Graphical lasso [3, 4] is a network estimation method that provides an interpretable sparse inverse covariance matrix due to the ℓ_1-norm. Time varying graphical lasso (TVGL) [5] is a network estimation method that takes time information into account. Although this method can find change points by comparing the network structure before and after a change, it can't find clusters. Toeplitz inverse covariance-based clustering (TICC) [6] and time adaptive Gaussian model (TAGM) [7] are clustering methods based on network structure. TICC uses Markov random fields (MRF) and Toeplitz matrices to capture the inherent relationships among variables. TAGM is a fusion of a hidden

Markov model (HMM) and a Gaussian mixture model (GMM). These methods find clusters depending on the network structure of each subsequence. This provides clusters with interpretability and allows us to discover patterns that other traditional clustering methods are unable to find. Both incorporate a graphical lasso to capture the interaction between variables but require the number of clusters to be specified as prior information. Consequently, only our approach satisfies the need for interpretability and find the optimal number of clusters automatically.

Contributions. The main contribution of this work is the concept and design of NGL, which has the following desirable properties:

1. **Interpretable**: NGL provides the underlying graphical structures and cluster assignments in data, which help us to interpret important relationships between variables.
2. **Automatic**: We formulate data encoding schemes to reveal distinct patterns/clusters, each of which captures the network structure. The proposed method requires no parameter tuning to specify the number of clusters.
3. **Accurate**: NGL is a simple yet powerful algorithm for time-series segmentation using a graphical lasso, which outperforms state-of-the-art competitors in terms of accuracy.

2. Preliminary

In this paper we investigate an automatic network structure clustering for large multidimensional time-series data. We will describe a few key concepts and background materials in this section.

2.1. Problem definition

Consider a set of p-dimensional time-series data consisting of T sequential bundle observations, $X = \{x_1, x_2, ..., x_T\}$ and there are $|x_i| \geq 1$ different observations at each time i. $x_i \in \mathbb{R}^p$ is the i-th multidimensional bundle observation, and each bundle observation vector is sampled from any K multivariate normal distributions $x_i \sim N(0, \Sigma_k)$. The network structure is equal to the graph structure, in which a given $x_i \sim N(0, \Sigma_k)$, each variable represents a node, and the covariance matrix Σ_k forms an edge. Our goal is to find the cluster assignments of these T bundle observations into K clusters $\mathcal{F} = \{\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_K\}$, where $\mathcal{F}_k$ is a cluster assignment set of $X_k \subset X\,(s.t.\,k = 1, 2, ..., K)$ represented by a set of K matrices, i.e., $\Theta = \{\theta_1, \theta_2, ..., \theta_K\}$. Therefore, letting $M_k = \{\theta_k, \mathcal{F}_k\}$ be a compact description for X_k in the k-th cluster, the full parameter set that we want to estimate consists of $M = \{M_1, M_2, ..., M_K\}$ and the number of clusters K.

2.2. Graphical lasso problem

We first consider the static inference that estimates θ_i. The optimization problem is written as follows:

$$\text{minimize}_{\theta_i \in S^p_{++}} - ll(x_i, \theta_i) + \lambda ||\theta_i||_{od,1},$$
$$ll(x_i, \theta_i) = |x_i|(\log\det\theta_i - Tr(S_i\theta_i)),$$

where S_i is the empirical covariance $(1/|x_i|)\sum_{j=1}^{|x_i|} x_j x_j^T$, x_j are the different samples, S^p_{++} is the space of the positive definite matrices, λ determines the sparsity level of the network, and $\|\cdot\|_{od,1}$ is the off-diagonal ℓ_1-norm. This is a convex optimization problem, which imposes the ℓ_1-norm restriction.

2.3. TVGL problem

To infer a time-varying sequence of networks, TVGL [5] extends the above approach, which is designed to infer a set of inverse covariance matrices Θ. TVGL solves the problem below:

$$\text{minimize}_{\theta_i \in S^p_{++}} \sum_{i=1}^{T} -ll(x_i, \theta_i) + \lambda ||\theta_i||_{od,1} + \beta \sum_{i=2}^{T} \psi(\theta_i - \theta_{i-1}),$$

where β determines how strongly correlated neighboring covariance estimations should be. The penalty function ψ encourages similarity between θ_i and θ_{i-1}. Different types of ψ allow us to enforce different restrictions in the time-varying similarity. This problem is solved by employing the alternating direction method of multipliers (ADMM) [8], which is a well-established method for solving the convex optimization problem. For more details, please see, e.g., [5]. Although TVGL can find a changing point by comparing θ_i and θ_{i-1}, it cannot find a cluster simultaneously. Throughout this paper our method uses TVGL to optimize the graphical lasso problem.

3. Algorithms

The previous section described how to estimate a set of inverse covariance matrices Θ. Now the questions are (a) how to describe the model, (b) how to find optimal cut points, and (c) how to assign segments to optimal clusters. There are three main ideas behind our model:

1. Model description cost: We use the minimum description length (MDL) principle as a model selection criterion for choosing between alternative segmentation and cluster descriptions. We

propose a novel cost function to estimate the description cost of the graphical lasso model.

2. CutPointSearch: We modify the generic bottom-up algorithm [9] to enhance its ability to handle time-series data. Initially we adopt short segments and iteratively merge with an adjacent pair that satisfies the cost restriction.

3. NGL: We use the EM algorithm to cluster the segments obtained by CutPointSearch while determining the optimal number of clusters automatically.

3.1. Model description cost

The MDL explains the model in a parsimonious way that calculates the required number of bits. Thus, it follows the assumption that the more we can compress the data, the more we can generalize its underlying structures. In a nutshell, we want to find the minimum number of graphical lasso models needed to express the data. The goodness of the model M can be described as follows:

$$< X; M > = < M > + < X|M >, \qquad (1)$$

where $< M >$ shows the cost of describing the model M, and $< X|M >$ represents the cost of describing the data X given the model M.

Model Coding Cost. The description complexity of model M is the sum of the following elements: The number of clusters K requires $\log^*(K)$. [1] The total number of observations of each cluster requires $\sum_{k=1}^{K} \log^*(|\mathcal{F}_k|)$. The mean value of each cluster which has a size $p \times 1$, requires $\sum_{k=1}^{K}(p \times c_F)$. The inverse covariance matrix of each cluster which has a size $p \times p$, requires $\sum_{k=1}^{K} |\theta_k|_{\neq 0}(2\log(p)+c_F)+\log^*(|\theta_k|_{\neq 0})$, where $|\cdot|_{\neq 0}$ describes the number of non-zero elements in a matrix and c_F is the floating point cost. [2]

Data Coding Cost. Given a model M, encoding cost of the data X using Huffman coding [10] is computed by: $< X|M >= \sum_{k=1}^{K} ll(X_k, \theta_k)$. Our next goal is to find the best model M that minimizes Equation (1).

3.2. Automatic cut point detection

So far, we have described how we calculate the MDL cost for our model. The next question is how to find optimal cut points that minimize MDL cost efficiently; we still have numerous candidates with which to merge to summarize similar subsequences into a compact model, and thus we modify the bottom-up algorithm to prevent a pattern explosion. We answer this question in two steps, MergeSegment and CutPointSearch.

[1]Here, $\log^*$ is the universal code length for integers
[2]We used 4×8 bits in our setting

3.2.1. MergeSegment (inner loop)

Assuming that neighboring segments tend to belong to the same cluster, we update cut points through MergeSegment. We consider given cut points $cp = \{c_0, c_1, ..., c_m\}$, and a set of inverse covariance matrices at each segment $\Theta_S = \{\theta_{,c_0}, \theta_{c_0,c_1}, ..., \theta_{c_m,}\}$, where the number of segments is $m+1$. And the set of inverse covariance matrices at each segment, which consists of only even/odd-numbered cut points $\Theta_E = \{\theta_{,c_0}, \theta_{c_0,c_2}, ...\}$ and $\Theta_O = \{\theta_{,c_1}, \theta_{c_1,c_3}, ...\}$. X_{c_i,c_j}, M_{c_i,c_j}, and θ_{c_i,c_j} are the data, model, and inverse covariance matrix from cut points c_i to c_j. Our goal is to determine if a segment should be merged with its neighboring segment. As shown in Figure 1, we have three candidates as updated cut points: (a) Solo has three segments all separated, (b) Left and (c) Right have two segments in which one side is merged. We compare the MDL costs using Equation (1), in these three cases, (a) vs. (b) vs. (c), and select the best cut points so that they minimize the local MDL cost. For example, if (b) has the lowest cost, c_{i+2} is added to the updated cut points. If (a) has the lowest cost, there is no change from the previous cut points. We iterate this process throughout the whole sequence.

3.2.2. CutPointSearch (outer loop)

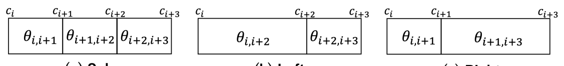
This algorithm finds the optimal cut points. We are now given bundle X and initial cut points cp. The user decides the interval of initial cut points. Since TVGL forces a time-varying similarity with neighboring network, we calculate Θ_S, Θ_O, and Θ_E using the TVGL graphical lasso optimization method. After obtaining each Θ, we run the MergeSegment algorithm to update the cut points. We iterate this process until the cut points are stable.

3.3. Automatic clustering: NGL

Now we have optimal cut points, which means that there are a limited number of segments that have enough samples with which to estimate the network structure. Next, we assign segments to a cluster and find the optimal number of clusters. As Algorithm 1 shows, we use the EM algorithm to classify each segment. For each iteration we vary $K = 1, 2, 3, ...$, and minimize the function below:

$$\arg\min \sum_{k=1}^{K} -ll(X_k, \theta_k) + \lambda||\theta_k||_{od,1}, \qquad (2)$$

In the E-step, we assign each segment to the optimal cluster, so that the log likelihood is maximized. In the M-step, we calculate the θ_k value of each cluster using a normal graphical lasso optimization algorithm. Until the cost function increases, we vary K so as to minimize the cost function.



(a) Solo (b) Left (c) Right

Figure 1: Illustration of the three candidates. We compare the each MDL costs of these candidates.

Algorithm 1 NGL(X, cp)

```
1: Input: Bundle $X$, initial cut point set $cp$
2: Output: Cluster parameters $\Theta$ and cluster assignments $\mathcal{F}$
3: $cp_{opt}$ = CutPointSearch$(X, cp)$;   $K = 1$;
4: while improving the total cost $< X; M >$ do
5:     $\Theta$ = ModelInitialization$(cp_{opt}, K)$;
6:     repeat
7:         $\mathcal{F}$ = AssignToCluster$(X, \Theta, cp_{opt})$;   /* E-step, Equation (2) */
8:         $\Theta$ = GraphicalLasso$(X, \mathcal{F})$;   /* M-step */
9:     until convergence;
10:    Compute $< X; M >$; // $M = \{\Theta, \mathcal{F}\}$
11:    $K = K + 1$;
12: end while
13: return $M = \{\Theta, \mathcal{F}\}$;
```

4. Experiments

We evaluate our method on both synthetic and real datasets.

4.1. Accuracy on synthetic data

In this section, we demonstrate the accuracy of NGL on synthetic data. We do so because there are clear ground truth networks with which to test the accuracy.

Experimental Setup. We randomly generate synthetic multidimensional data in $\mathbb{R}^5$, which follows a multivariate normal distribution $X \sim N(0, \theta^{-1})$. Each of the K clusters has a mean of $\vec{0}$, so that the clustering result is based entirely on the structure of the data. For each cluster, we generate a random ground truth inverse covariance as follows [11]:

1. Set $A \in \mathbb{R}^{5\times 5}$ equal to the adjacency matrix of an Erdős-Rényi directed random graph, where every edge has a 20% chance of being selected.
2. For every selected edge in A set $A_{ij} \sim$ Uniform$([-0.6, -0.3] \cup [0.3, 0.6])$. We enforce a symmetry constraint whereby every $A_{ij} = A_{ji}$.
3. Let $c = \lambda_{\min}(A)$ be the smallest eigenvalue of A, and set $\theta = A + (0.1 + |c|)I$, where I is an identity matrix.

We run our experiments on four different temporal sequences: "1,2,1","1,2,3,2,1","1,2,3,4,1,2,3,4","1,2,2,1,3,3,3,1". We generate each dataset 10 times and report the mean and standard deviation of the macro-F1 score.

Baseline Methods. We compare our method to three state-of-the-art methods and one ablation method. TICC [6] and TAGM [7] take network structure into account. Since both methods need to specify the number of clusters, we gave the true number of clusters only to these methods. AutoPlait [12] is multi-level HMM based automatic method for time-series clustering. NGL no-cps is our NGL method without CutPointSearch. Our method, including NGL no-cps, requires us to specify initial cut points. We set its interval at every 5 points throughout the synthetic experiments.

Clustering Accuracy. We set each segment in each of the examples to have 100 observations in $\mathbb{R}^5$ (for example, "1,2,1" has a total of 300 observations). Table 1 shows the clustering accuracy for the macro-F1 scores for each dataset. As shown, NGL significantly outperforms the baselines. Our method consistently achieves the highest accuracy and lowest standard deviation. AutoPlait does not consider network structure, so it does not find any clusters. Although we gave the true number of clusters K to TICC and TAGM, the average accuracy of our method is more than 10% higher. NGL no-cps shows that finding a large segment by CutPointSearch has meaning of grouping adjacent observations into the same cluster.

Effect of Total Number of Samples. We next focus on the number of samples required for each method to accurately find clusters. We take the "1,2,3,4,1,2,3,4" example and vary the number of samples. As shown in Figure 2, our method outperforms the baselines for almost all segment lengths. Our method has a constantly high average, even for relatively small segment lengths. This is because our CutPointSearch algorithm correctly find cut points even if the sample size is small.

4.2. Case study

Here, we show that our NGL provides an interpretable result with real-world financial data. In general, stocks, bonds, and currency prices are correlated. By examining historical financial data, we can infer a financial network structure to reveal the relationships between them. We use hourly currency exchange rate data [3] of AUD/USD, EUR/USD, GBP/USD, and USD/CAD from 2005 to 2018. Assuming that the underlying network structure is consistent for a week, we normalized the data for each week. We also set the initial cut points at a week to capture the weekly correlation trend. The top of Figure 3 shows the clustering result obtained with NGL. During the global financial crisis (from mid-2007 to early 2009), we found that the network structure changed. There are abrupt changes on 2016/5/16 $\sim$ 2016/6/5, the bottom of Figure 3 shows how the correlation changed during this period. As we can see, a correlation related to the United

[3]https://github.com/FutureSharks/financial-data

Table 1
Macro-F1 score of clustering accuracy for four different temporal sequences, comparing NGL with state-of-the-art methods (higher is better).

Model	**NGL**	TAGM (KDD'21)	TICC (KDD'18)	AutoPlait (SIGMOD'14)	NGL no-cps
1,2,1	$\mathbf{0.93 \pm 0.05}$	0.83 ± 0.25	0.85 ± 0.26	0.67	0.62 ± 0.13
1,2,3,2,1	$\mathbf{0.96 \pm 0.03}$	0.74 ± 0.21	0.89 ± 0.18	0.40	0.66 ± 0.15
1,2,3,4,1,2,3,4	$\mathbf{0.94 \pm 0.03}$	0.78 ± 0.26	0.82 ± 0.21	0.25	0.66 ± 0.11
1,2,2,1,3,3,3,1	$\mathbf{0.93 \pm 0.05}$	0.89 ± 0.17	0.83 ± 0.26	0.38	0.62 ± 0.07

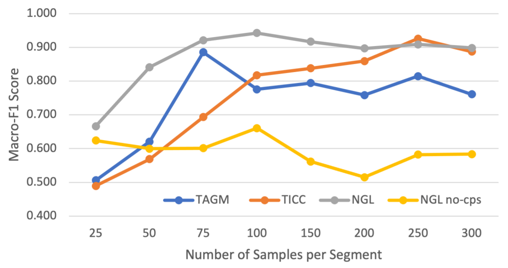


Figure 2: Plot of clustering accuracy macro-F1 score vs. number of samples for NGL and two other state-of-the-art methods.

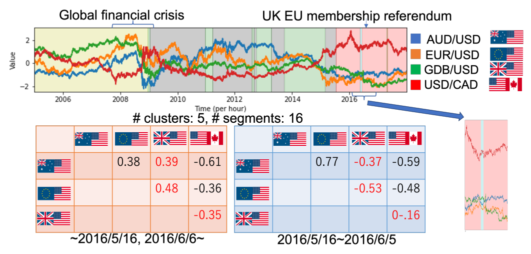


Figure 3: Clustering result of NGL using currency datasets.

Kingdom changed significantly. This was in response to the United Kingdom European Union membership referendum on 2016/6/23, which may have caused public concern.

5. Conclusion and Future work

In this paper, we presented NGL, which is an interpretable clustering algorithm. We focused on the problem of the interpretable clustering of multidimensional time-series data with underlying network structures. Our proposed NGL indeed exhibits all the desirable properties; it is *Interpretable* and *Automatic* and *Accurate*.

In future work, we will focus on the following direction: **Online learning.** In several situations, network inference needs to operate in an online fashion. And to the best of our knowledge, no study has dealt with online clustering based on network structure. In this context, we will develop an extension of our methods by utilizing the novel sliding window and bottom-up (SWAB) algorithm [9].

Acknowledgments

The authors would like to thank the anonymous referees for their valuable comments and helpful suggestions. This work was supported by JSPS KAKENHI Grant-in-Aid for Scientific Research Number JP20H00585, JP21H03446, JP22K17896, NICT 21481014, MIC/SCOPE 192107004, JST-AIP JPMJCR21U4, ERCA-Environment Research and Technology Development Fund JPMEERF20201R02,

References

[1] C. Miyajima, Y. Nishiwaki, K. Ozawa, T. Wakita, K. Itou, K. Takeda, F. Itakura, Driver modeling based on driving behavior and its evaluation in driver identification, IEEE 95 (2007) 427–437.

[2] F. Tomasi, V. Tozzo, A. Barla, Temporal pattern detection in time-varying graphical models, in: ICPR, 2021, pp. 4481–4488. doi:10.1109/ICPR48806.2021.9413203.

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

[4] F. Tomasi, V. Tozzo, S. Salzo, A. Verri, Latent variable time-varying network inference, in: KDD, 2018, pp. 2338–2346. URL: https://doi.org/10.1145/3219819.3220121. doi:10.1145/3219819.3220121.

[5] D. Hallac, Y. Park, S. P. Boyd, J. Leskovec, Network inference via the time-varying graphical lasso, in: KDD, 2017, pp. 205–213. URL: https://doi.org/10.1145/3097983.3098037. doi:10.1145/3097983.3098037.

[6] D. Hallac, S. Vare, S. P. Boyd, J. Leskovec, Toeplitz inverse covariance-based clustering of multivariate time series data, in: KDD, 2017, pp. 215–223. URL: https://doi.org/10.1145/3097983.3098060. doi:10.1145/3097983.3098060.

[7] V. Tozzo, F. Ciech, D. Garbarino, A. Verri, Statistical models coupling allows for complex local multivariate time series analysis, in: KDD, 2021, pp. 1593–1603. URL: https://doi.org/10.1145/3447548.3467362. doi:10.1145/3447548.3467362.

[8] S. P. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers, Found. Trends Mach. Learn. 3 (2011) 1–122. URL: https://doi.org/10.1561/2200000016. doi:10.1561/2200000016.

[9] E. J. Keogh, S. Chu, D. M. Hart, M. J. Pazzani, An online algorithm for segmenting time series, in: Proceedings of the 2001 IEEE International Conference on Data Mining, 29 November - 2 December 2001, San Jose, California, USA, IEEE Computer Society, 2001, pp. 289–296. URL: https://doi.org/10.1109/ICDM.2001.989531. doi:10.1109/ICDM.2001.989531.

[10] C. Böhm, C. Faloutsos, J.-Y. Pan, C. Plant, Ric: Parameter-free noise-robust clustering, TKDD 1 (2007) 10–es.

[11] K. Mohan, P. London, M. Fazel, D. Witten, S.-I. Lee, Node-based learning of multiple gaussian graphical models, J. Mach. Learn. Res. 15 (2014) 445–488.

[12] Y. Matsubara, Y. Sakurai, C. Faloutsos, Autoplait: Automatic mining of co-evolving time sequences, in: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, SIGMOD '14, Association for Computing Machinery, New York, NY, USA, 2014, p. 193–204. URL: https://doi.org/10.1145/2588555.2588556. doi:10.1145/2588555.2588556.