Clustered Fused Graphical Lasso

Yizhi Zhu
yzhu44@illinois.edu

Oluwasanmi Koyejo
sanmi@illinois.edu

Abstract

Estimating the dynamic connectivity structure among a system of entities has garnered much attention in recent years. While usual methods are designed to take advantage of temporal consistency to overcome noise, they conflict with the detectability of anomalies. We propose *Clustered Fused Graphical Lasso* (CFGL), a method using precomputed clustering information to improve the signal detectability as compared to typical Fused Graphical Lasso methods. We evaluate our method in both simulated and real-world datasets and conclude that, in many cases, CFGL can significantly improve the sensitivity to signals without a significant negative effect on the temporal consistency.

1 INTRODUCTION

In recent years, undirected graph models have become a popular topic in machine learning. In an undirected graphical model, vertices represent entities in the system, and edges represent bi-directional effects between entities. The inverse covariance matrix is the preferred estimator for such structures since it indicates partial correlations, i.e., an off-diagonal entry is zero if and only if the entities of the corresponding column and row are conditionally independent given all the other entities. Therefore, two adjacent vertices in an estimated network correspond to a non-zero off-diagonal entry and a direct dependency. One of the most popular methods for estimating the sparse precision matrix is *Graphical Lasso* (Glasso) [Friedman et al., 2008], which assumes the connectivity structure is static. However, this assumption is not satisfied in many fields like functional MRI [Monti et al., 2014], financial markets [Namaki et al., 2011], or social network analysis [Ahmed and Xing, 2009]. In such cases, data comes from a time series of collections, and the underlying structures are usually assumed to be non-static across time. Consequently, estimating dynamic networks at each time point becomes necessary in order to better understand the complex systems.

Compared to the static case, fewer observations at each time point are available in dynamic estimation. The lack of samples leads to higher level of noise, and thus introduces additional difficulty in estimation. Temporal consistency is a natural assumption with time-varying networks, based on the idea that in most cases, only few changes should occur between consecutive networks. Given this assumption, one may like to place an additional penalty on the difference of neighboring networks. *Fused Graphical Lasso* (Fused Glasso) achieves this using an element-wise l_1 penalty, and it has become the default choice for many studies in the structure estimation field [Monti et al., 2014, Hallac et al., 2017, Danaher et al., 2014, Ahmed and Xing, 2009].

Several Fused Glasso based algorithms have been proposed in the literature on time varying network estimation, and they all have some issues with change detection. SINGLE [Monti et al., 2014] avoids accurate change detection by assuming temporal homogeneity (i.e., small and slow changes) on functional MRI data. It uses Fused Glasso on sample covariance estimates, which are smoothed using a Gaussian kernel, so that all abrupt changes are transferred into trends. In another study [Gibberd and Nelson, 2017], the ability to recover change points is specifically targeted. *Grouped Fused Graphical Lasso* (GFGL) uses a group $l_{2,1}$ smoothing. The drawback is that compared to Fused Glasso results, GFGL has performance loss on static periods on a similar scale as the performance gain at change points. Time-Varying Graphical Lasso (TVGL) [Hallac et al., 2017] proposes a general framework, allowing various penalty functions to be applied to fit different situations. But again, it is difficult for a single penalty function to satisfy

both temporal consistency and temporal diversity. More importantly, estimating dynamic functional structures is an unsupervised task. Thus, it is usually impossible to know the correct situation beforehand or to objectively compare methods with different penalties.

In this work, we propose the *Clustered Fused Graphical Lasso* (CFGL) method, obtaining good detectability of changing events and taking care of temporal consistency. Motivated by the property that the thresholded hierarchical clustering is closely related to the connected components of Glasso estimated graphs [Mazumder and Hastie, 2012], we make the key observation that this clustering information indicates evidence of structure-change events and can be a reasonable heuristic. We propose a clustering framework to enhance the evidence of changes. CFGL incorporates the precomputed information into the smooth penalty so that local structures are free to detect change points.

Section 2 formulates the problem and briefly reviews the related background on Graphical Lasso and Fused Graphical Lasso estimation. Section 3 proposes the CFGL method and clarifies the algorithm details. Section 4 compares CFGL to existing methods in three different simulations. Section 5 evaluates CFGL in real cases. Section 6 concludes the paper and talks about future extensions.

2 BACKGROUND

2.1 PROBLEM DEFINITION

Say there is a sequence of multivariate observations at time points $t_1 \leq \cdots \leq t_T$. At each time t_i, $n_i \geq 1$ observation vectors form $X_i = \{x_i^1, ..., x_i^{n_i}\} \in \mathcal{R}^{n_i \times p}$ with $x_i \sim \mathcal{N}(0, \Sigma_i)$. We would like to infer the underlying connectivity structures across time. Based on the aforementioned precision matrix properties, an equivalent problem is to estimate the corresponding precision matrices $\{\Theta_i\} = \{\Theta_1, ..., \Theta_T\}$, one at each time point.

2.2 GRAPHICAL LASSO

We start from the static case, $T = 1$. An assumption on the number of dependencies (i.e., sparsity) is usually applied, so that only a subset with most dependencies is chosen. This would require placing a prior on the parameters to induce additional zeros on the off-diagonal entries of $\{\Theta_i\}$.

$$\arg\min_{\Theta} -l(\Theta) + \lambda_1 \left\|\Theta\right\|_1 \tag{1}$$

The equation (1) is *Graphical lasso* [Friedman et al., 2008], which is known to be one of the most effective methods for this problem [Wang et al., 2012]. The empirical covariance S is defined to be $\frac{1}{n}\sum_{i=1}^{n} x_i x_i^T$, and then the log likelihood $l(\Theta)$ is

$$l(\Theta_i) = \log\det(\Theta_i) - trace(S_i\Theta_i). \tag{2}$$

Minimizing $-l(\Theta)$ would encourage Θ to be close to the inverse covariance S^{-1} [Yuan and Lin, 2007]. The λ_1 in Equation (1) is a non-negative tuning parameter to trade off the sparsity and likelihood. $\left\|\Theta\right\|_1$ is defined to be the element-wise l_1 norm of Θ.

2.3 FUSED GRAPHICAL LASSO

In the case of $T > 1$, to estimate time series of graphs, we seek to take advantage of neighborhood information indicating temporal consistency, i.e., assume structures are similar to their neighbors. Previous methods [Danaher et al., 2014, Monti et al., 2014, Yang et al., 2015, Hallac et al., 2017] implement this assumption by adding an additional penalty to encourage smoothness. In particular, SINGLE [Monti et al., 2014], and l_1-penalized TVGL [Hallac et al., 2017] define this penalty to be an element-wise l_1 norm of the difference between consecutive estimations:

$$\arg\min_{\{\Theta_i\}} \sum_{i=1}^{T} -l(\Theta_i) + \lambda_1 \sum_{i=1}^{T} ||\Theta_i||_1 + \lambda_2 \sum_{i=2}^{T} \|\Theta_i - \Theta_{i-1}\|_1 \tag{3}$$

Equation (3) is usually called *Fused Graphical Lasso* (Fused Glasso), due to its relationship to its vector regularization analogue – *Fused Lasso* [Tibshirani et al., 2005], for estimating a sparse time-varying vector.

The variational norm in Fused Lasso is effective for introducing smoothness, but Qian and Jia [2012] prove that Fused lasso can recover exact patterns only if there are no consecutive change points on the timeline and all changes keep switching directions. Many real-world signal patterns obviously do not satisfy these conditions. In addition, even with the aforementioned conditions satisfied, simulated experiments on Fused Glasso show large F_1 score performance drop in the vicinity of the only change point [Gibberd and Nelson, 2017]. In response, we propose the CFGL to improve signal recovery.

3 CLUSTERED FUSED GRAPHICAL LASSO

3.1 PROPOSED METHOD

Instead of assigning a uniform smoothing weight on all the entries of all precision matrices, we explore methods to distinguish between stable and non-stable connections

in the network and only apply smooth penalties on stable connections.

Mazumder and Hastie [2012] proves the close relation between connected components of the thresholded sample covariance graph and connected components of the Glasso-estimated graph. Tan et al. [2015] extends the conclusion and proves the following theorem, which states the relation between Glasso and clustering results.

Denote $|S|$ as the matrix of element-wise absolute values of S, i.e., $|S|^{k,l} = |S^{k,l}|$, where S is the normalized $p \times p$ covariance matrix.

Theorem 3.1. Let $C^1, \ldots, C^K$ denote the clusters that result from performing *Single Linkage Hierarchical Clustering* (SLC) using similarity matrix $|S|$ and cutting the resulting dendrogram at a height of $0 \leq \lambda_1 \leq 1$. Let $D^1, ..., D^R$ denote the connected components of the graphical lasso solution with tuning parameter λ_1. Then, $K = R$, and there exists a permutation π such that $C^k = D^{\pi(k)}$ for $k = 1, ..., K$.

Following Theorem 3.1, it is clear that the regularization parameter λ_1 is the threshold to define connected components in the Glasso solution. We note that other clustering algorithms like *average linkage clustering* (ALC) are also used as alternative methods in Tan et al. [2015]. In the following sections, we will use *clusters* and *connected components* interchangeably. We also assume that λ_1 is a good threshold resulting in clustering with reasonable accuracy.

Define $V = \{v_1, ..., v_p\}$ as the vertex set representing the entities, and $C_i = \{C_i^1, ..., C_i^K\}$ as the clustering result on $|S_i|$ with parameter λ_1, where $C_i(v_k)$ is the cluster label of v_k. We construct an undirected graph $G_i = (V_i, E_i)$, where E_i is represented in a similar form to adjacency matrix. $E_i^{k,l} = 1$ if and only if there is a path between v_k and v_l, i.e.,

$$E_i^{k,l} = \mathbb{1}_{\{C_i(v_k) = C_i(v_l)\}}, \tag{4}$$

where $\mathbb{1}$ is the indicator function.

If $E_i^{k,l} = 0$, v_k and v_l have zero partial correlation at time t_i and thus $\Theta_i^{k,l} = 0$. If $E_i^{k,l} \neq 0$, v_k and v_l have non-zero partial correlation on intuition and thus $\Theta_i^{k,l} \neq 0$.

Given consecutive clustering results C_{i-1} and C_i, we want to use the evidence of partial correlation change on between Θ_{i-1} and Θ_i, and define the weight matrix $W_i \in \{0, 1\}^{p \times p}$ to help decide whether to apply the l_1 smooth penalty on each entry.

- If $E_{i-1}^{k,l} = E_i^{k,l}$, v_k and v_l both have either zero or non-zero partial correlation in time t_{i-1} and t_i, and we set $W_i^{k,l} = 1$.

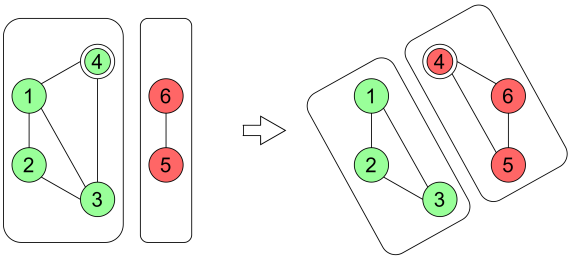


Figure 1: Vertex 4 detaches from the green cluster and merges to the red cluster.

- If $E_{i-1}^{k,l} \neq E_i^{k,l}$, we want the partial correlation between v_k and v_l to change freely. Thus $W_i^{k,l} = 0$.

A simple example is illustrated in Figure 1. At time t_{i-1} (left), $E_{i-1}^{4,l} = 1$ for $l \in \{1, 2, 3\}$ and $E_{i-1}^{4,l} = 0$ for $l \in \{5, 6\}$. At time t_i, $E_i^{4,l} = 0$ for $l \in \{1, 2, 3\}$ and $E_i^{4,l} = 1$ for $l \in \{5, 6\}$. Thus we set $W_i^{4,l} = 0$ for $l \in \{1, 2, 3, 5, 6\}$ to allow vertex 4 to freely change clusters.

Assuming the clustering threshold λ_1 is known, we propose the following steps:

1. Perform the chosen clustering method on each empirical covariance matrix S_i to obtain a sequence of cluster sets $C_1, C_2, ..., C_T$.
2. Let $\oplus$ denote the **XOR** logical operations, and define the weight matrix set $\{W_i\}$ as
$$W_i^{k,l} = 1 - \mathbb{1}_{\{E_{i-1}^{k,l} \oplus E_i^{k,l}\}}. \tag{5}$$
3. Apply these predefined weights and solve the following CFGL optimization problem:
$$\begin{aligned} \arg\min_{\{\Theta_i\}} \sum_{i=1}^{T} -l(\Theta_i) + \lambda_1 \sum_{i=1}^{T} ||\Theta_i||_1 \\ + \lambda_2 \sum_{i=2}^{T} ||(\Theta_i - \Theta_{i-1}) \circ W_i||_1, \end{aligned} \tag{6}$$
where the $\circ$ denotes the element-wise Hadamard product.

3.2 MORE STABLE CLUSTER CHANGES ACROSS TIME

The previously defined procedure in section 3.1 is sometimes sensitive to the choice of parameter λ_1. If λ_1 is small, clustering is sensitive to noise and very large clusters are always formed; if λ_1 is large, sparse clustering is achieved, but vertex pairs with true partial correlation are

likely to be overlooked. Even if λ_1 is within an acceptable range, the variations of values around the threshold may lead to grouped and detached clusters, introducing redundant switching. Therefore, no matter what value λ_1 is picked, the precomputed clustering change information tends to be noisy.

Since SLC is more unstable with noise and usually has undesirable chain structures [Hastie et al., 2009], Tan et al. [2015] use ALC instead of SLC for clustering on a single network. However, when merging individual and small clusters, ALC is similar to SLC and still suffers severely from noise. The frequent switching problem occurs on boundary values as well. We propose a framework to increase the stability of clustering changing across time (Algorithm 1) and make it applicable to most clustering algorithms.

The idea can be easily explained in the simplest case of hierarchical clustering. Consider there are two vertices, v_k and v_l, we propose to have two thresholds λ_1 and λ_1^*, with λ_1^* smaller than λ_1, i.e., $\lambda_1 = \lambda_1^* + \gamma$ and $\gamma > 0$.

- λ_1 is used to judge whether v_k and v_l should be grouped at time t_i if not grouped at time t_{i-1}.
- λ_1^* is used to judge whether they should be grouped again if they are grouped at time t_{i-1}.

In other words, we define the condition of v_k and v_l being clustered together as

$$\underset{C_i(v_k)=C_i(v_l)}{\text{Cond}} = \begin{cases} |S|_i^{k,l} \geq \lambda_1, & \text{if } i = 0 \text{ or } E_{i-1}^{k,l} = 0 \\ |S|_i^{k,l} \geq \lambda_1^*, & \text{if } E_{i-1}^{k,l} = 1 \end{cases} \tag{7}$$

To generalize the idea to complex cases (e.g., merging two clusters), Algorithm 1 is proposed. The input $\{S_i\}$ is the sequence of similarity matrices (normalized empirical covariance), γ can be understood as the gap or stabilizing parameter, and f is the clustering algorithm (i.e. represented as a function).

Algorithm 1 Stable Clustering Framework across Time

Input: $\{S_i\}$, γ, f
Output: $\{C_i\}$, $\{E_i\}$
1: $C_1 = f(|S_1|)$
2: Construct E_1 on C_1 using Equation (4)
3: **for** $i = 2$ to T **do**
4: $\quad \hat{S}_i = |S_{i-1}| + \gamma E_{i-1}$
5: $\quad C_i = f(\hat{S}_i)$
6: $\quad$ Construct E_i on C_i using Equation (4)
7: **end for**

The choice of γ depends on the clustering algorithm f, and also on the level of noise e we define (to maintain sparsity). Here we propose a heuristic choice for the hierarchical clustering case. Under the assumption that the previous clustering is accurate, there are two kinds of errors related to γ:

1. $|S_i|^{k,l} \geq \lambda_1 - \gamma$ with true $E_{i-1}^{k,l} = 1 \wedge E_i^{k,l} = 0$.
2. $|S_i|^{k,l} \leq \lambda_1 - \gamma$ with true $E_{i-1}^{k,l} = 1 \wedge E_i^{k,l} = 1$.

Let n be the sample size of each estimated S_i, the standard error of correlation coefficient $r = S_i^{k,l}$ is $se(r, n) = \sqrt{\frac{1-r^2}{n-2}}$. If we assume normally distributed error, the sampled correlation is $\bar{r} \sim \mathcal{N}(r, se(r, n)^2)$.

Let us assume the exact true correlation coefficient of all the k, l pairs is λ_1 if $E^{k,l} = 1$, and e if $E^{k,l} = 0$. Intuitively, we want $\lambda_1 - \gamma$ to be as far as possible from both λ_1 and e on their standard error normalized distances. Thus, we can heuristically estimate γ as:

$$\gamma = \frac{se(\lambda_1, n)(\lambda_1 - e)}{se(\lambda_1, n) + se(e, n)} \tag{8}$$

3.3 PARAMETER TUNING

All that remains is to tune the parameters λ_1 and λ_2. Following Hallac et al. [2017] and Monti et al. [2014], we use Akaike Information Criteria (AIC) to tune these hyperparameters. For a given pair (λ_1, λ_2), we define the AIC as:

$$AIC(\lambda_1, \lambda_2) = 2 \sum_{i=1}^{T} -l(\Theta_i) + 2K. \tag{9}$$

where the estimated degree of freedom K is slightly different from the definition in Tibshirani et al. [2005], and is given by:

$$K = \sum_{k,l} \sum_{i=2}^{T} \mathbb{1}_{\{(\Theta_i^{k,l} \neq 0 \,\oplus\, \Theta_{i-1}^{k,l} \neq 0) \,\wedge\, (\Theta_i^{k,l} \neq 0 \,\wedge\, W_i^{k,l} \neq 0)\}} \tag{10}$$

In equation (10), we do not penalize the changes unrelated to 0 since graphical lasso focuses more on the occurrence of edges. Term $W_i^{k,l}$ is added to avoid penalizing intentionally allowed changes.

Observe that a small λ_1 may result in huge clusters and $W_i^{k,l} \neq 0$ everywhere, which makes CFGL equivalent to Fused Glasso. In addition to a typical grid search with AIC score, CFGL requires λ_1 to be in a smaller predecided range. Therefore, we need to first tune a series of static graphical lasso (Static Glasso) with AIC to achieve an appropriate range of λ_1. Other methods can be used

to predefine the range, as long as they distinguish CFGL with typical Fused Glasso.

For a fixed sparse penalty λ_1, the clustering threshold can be either set to be λ_1 or slightly higher in order to compensate for the use of γ. The CFGL algorithm is described in Algorithm 2.

Algorithm 2 CFGL and Parameter Tuning

Input: $\{S_i\}$, f
Output: $\{\Theta_i\}$
Tune λ_1^* on static graphical lasso using AIC score
Obtain a range of λ_1 near the best λ_1^* from 1
for λ_1 among choices **do**
  for λ_2 among choices **do**
    Compute $\{E_i\}$ using $\{S_i\}, f$ and λ_1 via Algorithm 1
    Compute $\{W_i\}$ using $\{E_i\}$ and equation (5)
    Obtain $\{\Theta_i\}_{\lambda_1,\lambda_2}$ that minimizes equation (6)
    Compute AIC score of $\{\Theta_i\}$
  end for
end for
return $\{\Theta_i\}_{\lambda_1,\lambda_2}$ which minimizes the AIC score

3.4 OPTIMIZATION ALGORITHM

We use Alternating Directions Method of Multipliers (ADMM) [Boyd et al., 2011] algorithm to solve the optimization problem. Firstly, define the problem as:

$$\begin{aligned}\underset{\{\Theta_i\},\{Z_i\}}{\text{minimize}} &\sum_{i=1}^{T} -l(\Theta_i) + \lambda_1 \sum_{i=1}^{T} \|Z_i\|_1 \\ &+ \lambda_2 \sum_{i=2}^{T} \|(Z_i - Z_{i-1}) \circ W_i\|_1 \\ \text{subject to: } &\Theta_i = Z_i, \text{ for } i = 1, 2, 3, \ldots, T.\end{aligned} \tag{11}$$

The augmented Lagrangian corresponding to equation (11) is defined as:

$$\begin{aligned}\mathcal{L}_\rho(\{\Theta_i\},\{Z_i\},\{U_i\}) &= \sum_{i=1}^{T} -l(\Theta_i) \\ &+ \lambda_1 \sum_{i=1}^{T} \|Z_i\|_1 + \lambda_2 \sum_{i=2}^{T} \|(Z_i - Z_{i-1}) \circ W_i\|_1 \\ &+ \frac{\rho}{2} \sum_{i=1}^{T} \left(\|\Theta_i - Z_i + U_i\|_2^2 - \|U_i\|_2^2 \right),\end{aligned} \tag{12}$$

where $\{U_i\}$ are scaled dual variables, and ρ is a constant penalty parameter in ADMM which is usually set to one. Consequently, we get the update rule for U_i, Θ_i for $i = 1, \ldots, T$, and $\{Z_i\}$ in $j+1$th iteration:

$$\Theta_i^{j+1} = \underset{\Theta_i^{j+1}}{\arg\min} \frac{\rho}{2} \|\Theta_i - Z_i^j + U_i^j\|_2^2 - l(\Theta_i) \tag{13}$$

$$\{Z_i^{j+1}\} = \underset{\{Z_i^{j+1}\}}{\arg\min}\, \mathcal{L}_\rho(\{\Theta_i^{j+1}\},\{Z_i\},\{U_i^j\}) \tag{14}$$

$$U_i^{j+1} = U_i^j + \Theta_i^{j+1} - Z_i^{j+1} \tag{15}$$

Thus, the update step (13) is same in a typical fused graphical lasso, and the solution is discussed in detail by Monti et al. [2014] and Danaher et al. [2014]. For the Z update step (14), since all of these are element-wise operations, we can solve each $\{Z_i^{j+1}\}^{k,l}$ separately.

$$\begin{aligned}&\underset{\{Z_i^{j+1}\}^{k,l}}{\arg\min} \frac{\rho}{2} \sum_{i=1}^{T} \|\{\Theta_i^{j+1} - Z_i + U_i^j\}^{k,l}\|_2^2 \\ &+\lambda_1 \sum_{i=1}^{T} \|Z_i^{k,l}\|_1 + \lambda_2 \sum_{i=2}^{T} \|(Z_i^{k,l} - Z_{i-1}^{k,l}) \circ W_i^{k,l}\|_1\end{aligned}$$

Set $y_i = (\Theta_i^{j+1} + U_i^j)^{k,l}$ and $\beta_i = (Z_i)^{k,l}$. We get the following equation:

$$\begin{aligned}\mathbf{f}_{1,T}^*(\beta) = &\sum_{i=1}^{T} \frac{\rho}{2} \|y - \beta_i\|_2^2 + \lambda_1 \sum_{i=1}^{T} \|\beta_i\|_1 \\ &+ \sum_{i=2}^{T} \lambda_{i,i-1} \|\beta_i - \beta_{i-1}\|_1,\end{aligned}$$

where the $\lambda_{i,i-1}$ is defined to be $\lambda_2 W_i^{k,l}$.

Note that the nonzero value of $\lambda_{i,i-1}$ enforces β_i and β_{i-1} to be close, and a zero value in $\lambda_{i,i-1}$ splits the above equation into several smaller pieces. If $W_i^{k,l} \neq 0$ for all i, the equation is equivalent to a 1-dimensional FLSA as shown below, whose solver has been well discussed by Friedman et al. [2007], Hoefling [2010]:

$$\begin{aligned}\mathbf{f}_{1,T}(\beta) = &\sum_{i=1}^{T} \frac{\rho}{2} \|y - \beta\|_2^2 + \lambda_1 \sum_{i=1}^{T} \|\beta_i\|_1 \\ &+ \lambda_2 \sum_{i=2}^{T} \|\beta_i - \beta_{i-1}\|_1.\end{aligned}$$

If $W_i^{k,l} = 0$ for $i \in \{m_1, \ldots, m_D\}$, $m_0 = 1, m_1 \geq 2, m_D \leq T$ and $m_{D+1} = T$, solving the above problem is equivalent to separately solving $D+1$ independent 1-dimensional FLSA. In other words, β becomes the concatenation of $\{\beta^1, \ldots, \beta^{D+1}\}$, and

$$\beta^d = \underset{\beta}{\arg\min} f_{m_{d-1},m_d}(\beta).$$

It is shown in the supplement that $\{W_i\}$ does not introduce any additional complexity. Also, the update steps (13) and (14) can be easily parallelized, making the optimization algorithm very scalable.

4 SIMULATED EXPERIMENTS

To perform simulations, we first construct three groups of signal patterns, outlined in Figure 2, ranging from sudden stimuli to long-term switches. Group 1 has two short-term changes lasting for three time points. Group 2 has three short stimuli happening in only one time point. Signals in Group 3 have equal length and are sequentially distributed across time.

4.1 EXPERIMENT SETUP

We generate the simulated data from an Erdős–Rényi random graph $G = \{V, E\}$ under the controlled sparsity $|E| = 0.5\,|V|$. To form the precision matrix $\hat{\Theta}_i$ and sampled observation data, we use the following method, similar to Gibberd and Nelson [2017]

1. Set an empty $|V| \times |V|$ matrix and insert off-diagonal terms based on edges in G with values chosen from *Unif*$(0.6, 0.9)$.
2. Equally shift all the diagonal entries by a positive value so that the smallest eigenvalue is 0.1 to ensure positive semi-definiteness.
3. Normalize the matrix to have value 1 on diagonal.
4. Generate observation data $X_i = \{x_1^1, \ldots, x_1^{n_i}\}$ and $x_i^j \sim \mathcal{N}(0, \hat{\Theta}_i^{-1})$.

For all the three simulations datasets, we set $|V| = 25$. The generated precision matrices $\{\hat{\Theta}_i\}$ have off-diagonal terms with values around 0.5. There are total 30 time points, each with 25 observations, i.e., $T = 30$ and $n_i = 25$ for all i.

We compare to four state-of-the-art baseline methods, the static graphical lasso [Friedman et al., 2008], the fused graphical lasso from SINGLE [Monti et al., 2014], the l_1 penalized TVGL [Hallac et al., 2017], and the group fused graphical lasso (GFGL) [Gibberd and Nelson, 2017]. For Static Glasso, the graph at each time t_i is estimated independently. So we solve a total of T Glasso (1) problems to get T separate graphs. The degree of freedom K in Glasso is defined as the number of none zero entries [Tibshirani et al., 2005]. Although both SINGLE and l_1-TVGL have exactly the same fused graphical lasso objective in optimization, they use different optimization solvers which often result in different performance. We put both methods here as baselines and denote them as FGL-SINGLE and FGL-TVGL accordingly. Also, it is worth mentioning that CFGL shares a similar optimization solver as FGL-SINGLE.

We use four different versions of CFGL to compare to three baseline methods, listed in table 1. The Γ represents a rough estimation by applying equation (8) on AIC tuned Static Glasso λ_1. CFGL related methods are tuned using Algorithm 2. The four baseline methods are tuned by a typical grid search to minimize their correspondingly defined AIC score. Gibberd and Nelson [2017] tentatively propose a BIC score to tune GFGL in unsupervised tasks, but we find AIC generates slightly better results in our simulations. To eliminate the potential performance difference caused by tuning range, baseline methods are also tuned by Algorithm 2, and the result with better F_1-score is chosen. We repeat the whole process 10 times to reduce randomness, each time generating a new set of Erdős–Rényi graphs and observations. The averaged performance results are shown in Table 2.

Table 1: Notation of CFGL Related Methods

Notation	Clustering Method	Threshold
`CFGL-alc`	ALC	λ_1
`CFGL-slc`	SLC	λ_1
`CFGL-alc2`	ALC	$\lambda_1 + \Gamma/2$
`CFGL-slc2`	SLC	$\lambda_1 + \Gamma/2$

4.2 PERFORMANCE METRICS

We measure the performance of each method using three metrics: F_1-score, F_1-ratio, and edge deviation ratio.

4.2.1 F_1 Score

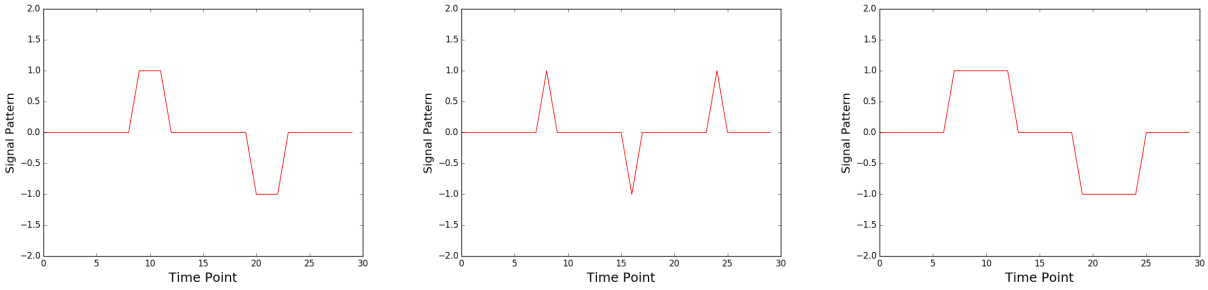
This measures how close the captured structures are to the true graphs. The F_1 Score shown in Table 2 is the averaged F_1 score across all the time points. Thus it provides an overview of performance for both static points and signals.

4.2.2 F_1 Ratio

We define F_1 ratio to be the average ratio between F_1 scores at the starting points of changing signals and the overall averaged F_1 score. F_1 ratio indicates the performance of accurate edge detection at change points.

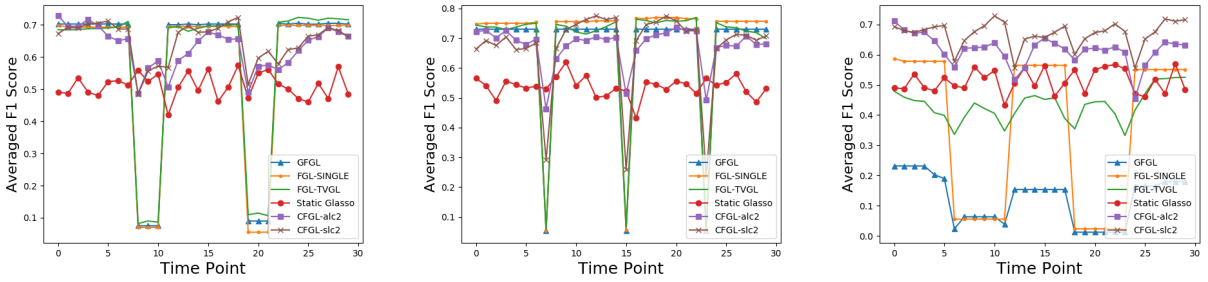
4.2.3 Edge Deviation (ED) Ratio

We define ED Ratio to be the average ratio between edge deviations (number of changing edges) at starting points of changing signals and the overall averaged edge deviations. Unlike F_1 ratio which pays more attention to the correctness, ED ratio focuses more on changing detecting and provides an unsupervised measure.



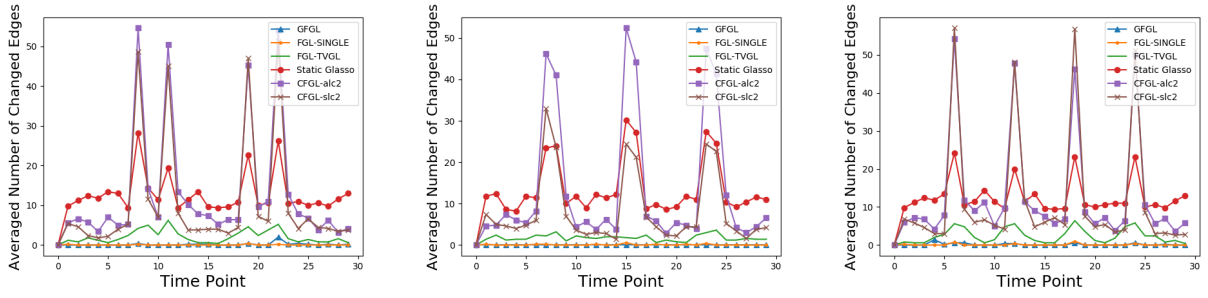
(a) Signal Group 1 (b) Signal Group 2 (c) Signal Group 3

Figure 2: Signal patterns with same numerical values have exactly same graph structure and ground truth precision matrix. Signal patterns with same absolute values but opposite signs have the same graph structure but opposite signs in the off diagonal values in precision matrix.



(a) F_1 Score Group 1 (b) F_1 Score Group 2 (c) F_1 Score Group 3

Figure 3: Each point on the curve represents the averaged F_1 score at that time point among 10 repetitions.



(a) Edge Changes Group 1 (b) Edge Changes Group 2 (c) Edge Changes Group 3

Figure 4: Each point on the curve represents the averaged number of changing edges between consecutive time points among 10 repetitions.

Table 2: Performance Comparison

Dataset	Signal Group1			Signal Group2			Signal Group3		
Metrics	F_1 Score	F_1 Ratio	ED Ratio	F_1 Score	F_1 Ratio	ED Ratio	F_1 Score	F_1 Ratio	ED Ratio
GFGL	0.580	0.143	2.6	0.662	0.081	0	0.129	0.147	3.6
FGL-SINGLE	0.570	0.110	**9**	**0.687**	0.076	**5.45**	0.400	0.112	**9.6**
FGL-TVGL	0.593	0.164	2.15	0.672	0.085	1.37	0.438	0.797	2.53
Static Glasso	0.526	**1.01**	2.04	0.545	**1.00**	2.02	0.523	**1.02**	1.93
CFGL-alc	0.631	0.848	3.43	0.630	0.800	2.81	0.618	0.934	3.40
CFGL-slc	0.555	0.115	2.14	**0.687**	0.078	2.8	0.626	0.932	5.18
CFGL-alc2	0.633	0.775	3.87	0.676	0.724	3.61	0.618	0.927	3.94
CFGL-slc2	**0.653**	0.768	4.55	0.670	0.392	3.35	**0.668**	0.882	4.90

4.3 SIMULATION RESULTS

CFGL-alc2 and CFGL-alc have stably good performance in all three metrics. The comparison between CFGL-slc2 and CFGL-slc indicates that SLC based clustering is more sensitive to the choice of thresholds.

Although Static Glasso always achieves good values on F_1 ratio and ED ratio, its low F_1 score implies not using neighborhood information. The performance of FGL-SINGLE shows that it tends to choose parameters which strongly highlight temporal consistency, resulting in overlooking signals. FGL-TVGL performs slightly better on signal detection, but both the metrics in Table 2 and the curves in Figures 3 and 4 show that it fails to take care of temporal consistency and diversity at the same time. For GFGL, although our simulation perfectly satisfies their assumption that change time points tend to group together and the other time points remain static, GFGL estimated structures do not reflect the changing periods properly. The static periods are also highly disturbed by long changing periods in signal group 3. This may be due to the difficulty finding good parameters for GFGL in unsupervised scenarios, which was mentioned by Gibberd and Nelson [2017] as well.

5 CASE STUDIES

In this section, we apply CFGL to more complicated real-world datasets to show how the CFGL method can be used to provide insights into real-world multivariate time series datasets.

5.1 EEG EYE STATE

Electroencephalography (EEG) is a medical imaging technique that reads scalp electrical activities generated by brain structures, and EEG measurements are commonly used in medical and research areas to infer brain activities [Teplan et al., 2002]. The dataset we use is the EEG eye state dataset [Roesler, 2013] from the UC Irvine repository. All data is from one continuous EEG measurement with the Emotiv EEG Neuroheadset. The eye state was detected via a camera during the measurement and added later manually after analyzing the video frames.

Table 3: Cross Validated Results on Estimated Structures

Notation	Cross Validated Accuracy
Static Glasso	0.6625
FGL-TVGL	0.800
FGL-SINGLE	0.6875
CFGL-alc	0.8375
CFGL-alc2	**0.8875**

In this experiment, we pick the first 2000 observations, and we group every 25 observations to form a time point, which is roughly 0.2 seconds. All the methods use the aforementioned tuning procedure. Considering there is no ground truth graph structure in EEG data, we treat the video captured eye state as a ground truth signal. We use the estimated structures as transformed features (i.e., edges present or not) and train a linear SVM model to predict the corresponding eye state. We observe that the prediction performance is related to the feature sparsity, and thus we tune the parameters so that all methods have similar sparsity (about 25% edges present). The result of 10 fold cross-validation is shown in Table 3. In addition, we provide an aligned comparison between the eye state and edge changes. Considering the uncontrollable brain activities and the latency between brains and eye states, we smooth the edge-change signals by the Gaussian kernel. As shown in Figure 5, CFGL performs very well for capturing eye state switching events.

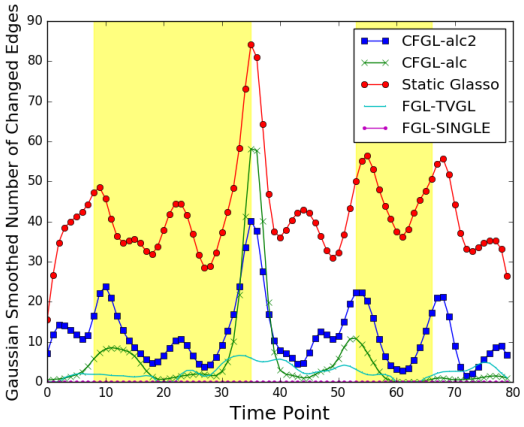


Figure 5: Edge changes are smoothed by Gaussian kernel. The yellow regions represent the period when subject closed his/her eyes.

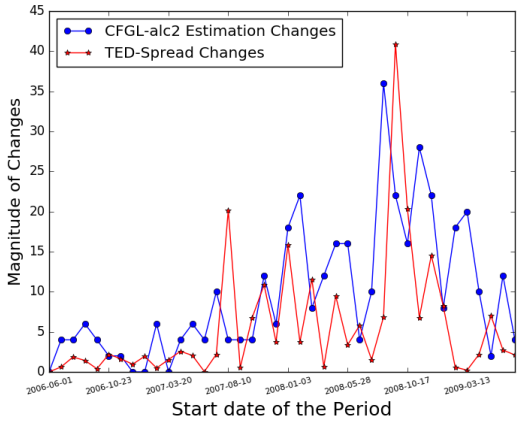


Figure 6: Edge Changes and TED Spread Changes

5.2 STOCK MARKET

In this experiment, we apply CFGL to the financial data in the stock market to explore the economic structures of stocks. These structures can be used to provide high-level understanding of company relations. In particular, the stock prices are natural multivariate time-series datasets, and they are also good indicators of company conditions. We assume each company's stock price is dependent on companies in the same or related fields, and is more likely to be independent to companies in unrelated fields. We pick 20 big companies, roughly 2 from each category of OS, Internet Service, PC, Auto, Restaurant, Finance, Energy, and Sales. Then we infer their structure changes during the global financial crisis of 2008[1]. We pick the dates starting from June 1st 2006 to August 4th 2009, total 800 days in the stock market, and use 20 days to form a time point so that each roughly represents a month.

TED spread is defined as the difference between the interest rates on interbank loans and on short-term U.S. government debt. It is usually treated as the indicator of perceived credit risk in the general economy [Boudt et al., 2017]. Thus we use the TED spread's changes as the reference of financial events. Figure 6 shows the comparison between the structural changes in estimated stock networks, and the TED spread changes between consecutive periods. It can be observed that the two curves follow similar patterns. There is a shift between the largest change of TED spread and stock market structure. We further investigated and found that the stock market started to drop in early September (around September 8th), which is exactly at the time of the blue peak but one time point earlier than the red peak. This result may indicate that TED spread has some latency for detecting stock market changes.

[1]Data freely available online from https://quantquote.com/historical-stock-data.

6 CONCLUSIONS AND FUTURE WORK

We propose *Clustered Fused Graphical Lasso* (CFGL) to improve the signal sensitivity of Fused Graphical Lasso (FGL). CFGL applies clustering based heuristic information on the smooth penalty so that temporal consistency and temporal diversity are simultaneously considered. Our experimental results show that the clustering information often makes CFGL more sensible for capturing signal changes, and CFGL outperforms FGL methods on datasets with time-varying underlying structures. The incorporated clustering information is independent of the smooth penalties. Therefore, there are many possible extensions either applying this information to other penalties, or setting up better methods for clustering.

Acknowledgements

We would like to thank to Microsoft Azure for computing resources.

References

Amr Ahmed and Eric P Xing. Recovering time-varying networks of dependencies in social and biological studies. *Proceedings of the National Academy of Sciences*, 106(29):11878–11883, 2009.

Kris Boudt, Ellen CS Paulus, and Dale WR Rosenthal. Funding liquidity, market liquidity and TED spread:

A two-regime model. *Journal of Empirical Finance*, 43:143–158, 2017.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

Patrick Danaher, Pei Wang, and Daniela M Witten. The joint graphical lasso for inverse covariance estimation across multiple classes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(2): 373–397, 2014.

Jerome Friedman, Trevor Hastie, Holger Höfling, Robert Tibshirani, et al. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

Alexander J Gibberd and James DB Nelson. Regularized estimation of piecewise constant Gaussian graphical models: The group-fused graphical lasso. *Journal of Computational and Graphical Statistics*, 26(3): 623–634, 2017.

David Hallac, Youngsuk Park, Stephen Boyd, and Jure Leskovec. Network inference via the time-varying graphical lasso. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 205–213. ACM, 2017.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning; Data Mining, Inference and Prediction*. Springer Verlag, New York, 2009.

Holger Hoefling. A path algorithm for the fused lasso signal approximator. *Journal of Computational and Graphical Statistics*, 19(4):984–1006, 2010.

Rahul Mazumder and Trevor Hastie. Exact covariance thresholding into connected components for large-scale graphical lasso. *Journal of Machine Learning Research*, 13(Mar):781–794, 2012.

Ricardo Pio Monti, Peter Hellyer, David Sharp, Robert Leech, Christoforos Anagnostopoulos, and Giovanni Montana. Estimating time-varying brain connectivity networks from functional MRI time series. *NeuroImage*, 103:427–443, 2014.

A Namaki, AH Shirazi, R Raei, and GR Jafari. Network analysis of a financial market based on genuine correlation and threshold method. *Physica A: Statistical Mechanics and its Applications*, 390(21-22):3835–3841, 2011.

Junyang Qian and Jinzhu Jia. On pattern recovery of the fused lasso. *arXiv preprint arXiv:1211.5194*, 2012.

Oliver Roesler. UCI machine learning repository, 2013. URL `https://archive.ics.uci.edu/ml/datasets/EEG+Eye+State`.

Kean Ming Tan, Daniela Witten, and Ali Shojaie. The cluster graphical lasso for improved estimation of Gaussian graphical models. *Computational statistics & data analysis*, 85:23–36, 2015.

Michal Teplan et al. Fundamentals of EEG measurement. *Measurement science review*, 2(2):1–11, 2002.

Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108, 2005.

Hao Wang et al. Bayesian graphical lasso models and efficient posterior computation. *Bayesian Analysis*, 7 (4):867–886, 2012.

Sen Yang, Zhaosong Lu, Xiaotong Shen, Peter Wonka, and Jieping Ye. Fused multiple graphical lasso. *SIAM Journal on Optimization*, 25(2):916–943, 2015.

Ming Yuan and Yi Lin. Model selection and estimation in the Gaussian graphical model. *Biometrika*, 94(1): 19–35, 2007.