การประมาณร่วมแบบจำลองเชิงกราฟเกรนเจอร์หลายแบบจำลองโดยใช้ฟังก์ชันลงโทษแบบไม่คอนเวกซ์

นายปรินทร มโนมัยเสาวภาคย์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2563

JOINT ESTIMATION OF MULTIPLE GRANGER GRAPHICAL MODELS USING NON-CONVEX PENALTY FUNCTIONS

Mr. Parinthorn Manomaisaowapak

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Electrical Engineering
Department of Electrical Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2020

Thesis Title	JOINT ESTIMATION OF MULTIPLE GRANGER GRAPHICAL MODELS USING NON-CONVEX PENALTY FUNCTIONS
By	Mr. Parinthorn Manomaisaowapak
Field of Study	Electrical Engineering
Thesis Advisor	Associate Professor Jitkomut Songsiri, Ph.D.

Accepted by the Faculty of Engineering, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

Dean of the Faculty of Engineering
. . . .
(Professor Supot Teachavorasinskun, D.Eng.)

THESIS COMMITTEE

. . . . Chairman
(Professor David Banjerdpongchai, Ph.D.)

. . . . Thesis Advisor
(Associate Professor Jitkomut Songsiri, Ph.D.)

. . . . External Examiner
(Tanagorn Jennawasin, Ph.D.)

ปรินทร มโนมัยเสาวภาคย์: การประมาณร่วมแบบจำลองเชิงกราฟเกรนเจอร์หลายแบบจำลองโดยใช้ฟังก์ชันลงโทษแบบไม่คอนเวกซ์. (JOINT ESTIMATION OF MULTIPLE GRANGER GRAPHICAL MODELS USING NON-CONVEX PENALTY FUNCTIONS) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : รศ. ดร. จิตโกมุท ส่งศิริ, 70 หน้า.

วิทยานิพนธ์นี้นำเสนอการเรียนรู้ร่วมแบบจำลองเชิงกราฟเกรนเจอร์ชนิดเบาบาง เพื่อหาความสัมพันธ์เชิงเหตุแบบเกรนเจอร์ชนิดร่วมและชนิดแตกต่างจากหลายอนุกรมเวลา การเรียนรู้ดังกล่าวสามารถนำไปใช้ในการวิเคราะห์ข้อมูลอนุกรมเวลาเอฟเอ็มอาร์ไอ เพื่ออนุมานโครงข่ายสมองที่มีร่วมกันของทุกบุคคลในกลุ่มตัวอย่าง หรือนำไปใช้ในการอนุมานความแตกต่างของโครงข่ายสมองในกลุ่มบุคคลที่มีอาการทางสมองที่แตกต่างกัน ความสัมพันธ์เชิงเหตุแบบเกรนเจอร์ระหว่างอนุกรมเวลาสามารถระบุได้จากตำแหน่งของศูนย์ร่วมในตัวแปรของแบบจำลองเวคเตอร์ถดถอยในตัว ดังนั้นวิทยานิพนธ์นี้จึงนำความสัมพันธ์ดังกล่าวมาใช้ในการประมาณร่วมของหลายแบบจำลอง ด้วยวิธีกำลังสองต่ำสุดแบบทำให้เป็นปกติด้วยนอร์มกลุ่มที่อยู่ในรูปแบบของฟังก์ชันลงโทษแลซโซชนิดกลุ่มหรือชนิดเชื่อมกัน ผลเฉลยของปัญหาดังกล่าวมีรูปแบบโครงข่ายความสัมพันธ์ที่แบ่งเป็นโครงข่ายร่วมและโครงข่ายเฉพาะตัวของแต่ละแบบจำลอง นอกจากนี้ เราใช้สมมติฐานของระดับความเบาบางในการกำหนดค่าถ่วงน้ำหนักสัมพัทธ์ในฟังก์ชันลงโทษและใช้นอร์มกลุ่มชนิดไม่คอนเวกซ์เพื่อเพิ่มความแม่นยำของวิธีที่นำเสนอในกรณีที่มีจำนวนข้อมูลน้อย การทดลองเชิงเลขบนข้อมูลที่จำลองขึ้นมาแสดงให้เห็นว่าวิธีที่นำเสนอมีประสิทธิภาพสูงกว่าในงานวิจัยที่ผ่านมา ในส่วนสุดท้าย วิทยานิพนธ์นี้ใช้วิธีที่นำเสนอในการวิเคราะห์ข้อมูลอนุกรมเวลาเอฟเอ็มอาร์ไอในระยะพักระหว่างกลุ่มของผู้เยาว์ที่มีโรคสมาธิสั้นและผู้เยาว์ที่มีการเติบโตปกติจากฐานข้อมูลเอดีเอชดี-200 เพื่ออนุมานความแตกต่างของโครงข่ายสมองเชิงประสิทธิภาพ ผลลัพธ์บ่งชี้ว่าความแตกต่างดังกล่าวส่วนมากอยู่ในบริเวณออร์บิโทฟรอนทัลและบริเวณที่เกี่ยวข้องกับระบบลิมบิกซึ่งสอดคล้องกับการศึกษาก่อนหน้าทั้งในด้านของงานวิจัยในเชิงคลินิกและงานวิจัยที่ใช้วิธีการวิเคราะห์ข้อมูล

ภาควิชา	วิศวกรรมไฟฟ้า	ลายมือชื่อนิสิต	....
สาขาวิชา	วิศวกรรมไฟฟ้า	ลายมือชื่ออ.ที่ปรึกษาหลัก	....
ปีการศึกษา	2563		

6270154921: MAJOR ELECTRICAL ENGINEERING

KEYWORDS: GRANGER CAUSALITY, EFFECTIVE BRAIN CONNECTIVITY, NON-CONVEX PENALTY, COMPOSITE PENALTY

PARINTHORN MANOMAISAOWAPAK : JOINT ESTIMATION OF MULTIPLE GRANGER GRAPHICAL MODELS USING NON-CONVEX PENALTY FUNCTIONS. ADVISOR : Assoc. Prof. Jitkomut Songsiri, Ph.D., 70 pp.

This thesis considers joint learning of multiple sparse Granger graphical models to discover underlying common and differential Granger causality (GC) structures across multiple time series. The proposed learning technique can be used in fMRI data analysis to infer a group-level brain network from a group of subjects or to uncover brain network differences among individuals with different brain conditions. By recognizing that the GC of a single multivariate time series can be characterized by common zeros of vector autoregressive (VAR) lag coefficients, a group norm penalty is included in joint regularized least-squares estimations of multiple VAR models. Group-norm regularizations based on a group- and fused-lasso penalty functions encourage the decomposition of multiple networks into a common GC structure, with other remaining parts defined in individual-specific networks. Prior information of sparsity patterns of desired GC networks are incorporated as relative weights, while a non-convex group norm in the penalty is proposed to enhance the accuracy of a network estimation in a low-sample setting. Extensive numerical results on simulations illustrated our method's improvements over existing sparse estimation approaches on sparse GC networks estimation. Our methods were also applied to available resting-state fMRI time series from the ADHD-200 data sets to learn the differences of effective brain connectivity between adolescents with ADHD and typically developing children. Our analysis revealed that parts of the causality differences between the two groups often resided in the orbitofrontal region and areas associated with the limbic system, which agreed with clinical findings and data-driven results in previous studies.

Department	: Electrical Engineering	Student's Signature	....
Field of Study	: Electrical Engineering	Advisor's Signature	....
Academic Year	: 2020		

Acknowledgements

First and foremost, I would like to convey my gratitude to my adviser for her patience and constructive and helpful feedback on my research. My adviser taught me a lot of things, including coding, research skills, what to read, and what to think. My colleagues, Poomipat, Tadchanon, Chantawit, Natthapol, and Satayu, have been really helpful and supportive. Without them, studying and researching here would be much more difficult. It was a pleasure to spend time with them. Finally, I would not have made it this far without the help of my family. I would like to thank my parents for their encouragement and support throughout my studies.

Without their support, this thesis would not have been possible.

CONTENTS

Page

Page

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

Chapter I

INTRODUCTION

The influences exerted by one region of the human brain on another are described as the effective brain connectivity or the causality flow within the human brain. Granger causality (GC) analysis is a model-based method that can reveal effective brain connectivity or causal interconnections among brain regions from neural activity data using vector autoregressive (VAR) models; see a review of connectivity inference in de Abril et al. (2018). Unfortunately, without prior information on the estimated network of interconnections, the resulting Granger graphical model (or GC network) is typically a dense estimate. Since it is difficult to make inferences about the brain's interactive structure from dense models, sparser solutions are more desirable, which can be achieved using regularization and Bayesian inference frameworks (de Abril et al., 2018). Sparse GC networks are studied in many variations for insignificant causality filtering, including lasso GC (Fujita et al., 2007), group lasso GC (Lozano et al., 2009), or truncated lasso (Shojaie and Michailidis, 2010). Recently, Bore et al. (2020) proposed to use a non-convex ℓ_q penalty with $0 < q < 1$ to improve GC estimation since it was theoretically superior to a lasso-type penalty in a linear regression context (Wen et al., 2018).

This thesis is concerned with the advancing of GC network estimations in two ways: simultaneously learning multiple GC networks, and constructing GC networks using a non-convex sparsity-inducing penalty. Simultaneous learning in the context of multiple sparse Gaussian graphical models currently exists as an estimation of the sparse inverse covariance matrix of random vectors, otherwise known as the graphical lasso problem (Friedman et al., 2007). A zero pattern of the inverse covariance, also known as the precision matrix, indicates conditional independence between random variables. This framework has been extended to joint Gaussian graphical model estimations with added prior knowledge on possible relationships among all models to force the models decomposed into two parts, a common and a differential part. The common part is defined as the common nonzeros in the inverse covariance matrices of all models, and the differential part refers to all other remaining nonzeros. Prior knowledge of common causality connections can be used to form a group lasso regularization to make all models have identical sparsity patterns (Liang et al., 2016; Ma and Michailidis, 2016), or a fused lasso to shrink the parameter differences among models (Danaher et al., 2014; Saegusa and Shojaie, 2016). Hara and Washio (2013) decomposed the precision matrices of multiple models into differential parts whose sparsity was promoted by a lasso, and a common part that were jointly regularized through a convex composite $\ell_{1,p}$ norm[1] for $p = 1, 2, \infty$. Other variants of single Gaussian graphical modeling approaches extended for multiple modeling also exploited similar lasso-type techniques; for example, these have involved a row and column inverse covariance estimation of the matrix Gaussian distribution (Huang and Chen, 2014), or

[1] The notation of $\ell_{p,q}$ refers to the composite of ℓ_q with ℓ_p norm.

the estimation of the inverse covariance and correlation matrices of Gaussian graphical models with explanatory variables (Huang et al., 2018). Recently, Yuan et al. (2021) proposed to use a joint penalty, weighted group lasso and weighted group fused lasso, in a joint Gaussian graphical model with a constraint that inverse covariance matrix is a Laplacian matrix.

Joint learning of Gaussian graphs has also been extended to use non-convex penalties, such as ℓ_0 to regularize non-zero elements (Tao et al., 2016), the $\ell_{1,1/2}$ norm (Guo et al., 2011), a general composite $\ell_{1,q}$ norm with $0 < q < 1$, truncated logarithmic penalty and inverse polynomial penalty (Chun et al., 2015). Despite several extensions of Gaussian graphical models to joint modeling, Gaussian graphs lack the core functionalities of capturing the temporal dependencies of time series and causality directions, issues which can be solved by multiple sparse GC modeling.

Current approaches to jointly estimate multiple GC graphs have estimating the common edges among all estimated networks forming *a common GC network*, and the differential edges that are unique to individual models forming *differential GC networks*. A group lasso is often applied to penalize parameters across models in the estimation of a common GC network (Songsiri, 2017; Gregorova et al., 2015). To learn both common and differential GC networks, several studies employed a combination of fused lasso and other lasso variants in regularized least-squares estimations (Skripnikov and Michailidis, 2019a; Wilms et al., 2018; Songsiri, 2015). The fused lasso encouraged some parameters to be identical across models, establishing the common network, while the other lasso-type penalty promoted zeros in each model's parameters in building the differential networks. However, because the VAR-based null GC of a single model contains all the *common zeros of all lag coefficients* (Lütkepohl, 2005), when the lasso-type regularization did not penalize all VAR-lag parameters as in Skripnikov and Michailidis (2019a); Wilms et al. (2018), the induced sparsity pattern did not directly reflect the null GC. Instead, VAR-lag matrices of an individual model should be penalized in groups, as was accomplished with a group-norm penalty in Songsiri (2015); Skripnikov and Michailidis (2019b); Gregorova et al. (2015). Finally, Skripnikov and Michailidis (2019b) proposed a two-stage procedure to jointly estimate multiple GC graphs using a group lasso penalty where common and differential GC networks were extracted in two separate stages, respectively.

Meanwhile, all of the aforementioned studies have relied on the group- and fused-lasso which are convex penalties. Applying non-convex penalties in applications of GC analysis is a relatively new approach, and only two recent works are worth noting. In one study, Bore et al. (2020) proposed using the non-convex $\ell_{1/2}$-norm penalty for estimating a single GC network, but without a group penalization of all VAR-lag parameters to relate null GC with the zero groups of parameters; this group-norm penalty ($\ell_{2,1/2}$) was an extension from the $\ell_{1/2}$ norm which had been shown to yield superior performances in recovering the true sparse solution over its convex counterpart (Hu et al., 2017). In the other study, the $\ell_{2,1/2}$ penalty was applied in Manomaisaowapak and Songsiri (2020) to estimate multiple VAR models having an identical GC structure. Despite a performance gain from the non-convex penalty under some settings, the formulation in Manomaisaowapak and Songsiri (2020) has room for improvement by introducing relative weights in the group penalty as a prior describing the edge strengths in the GC networks. This can be accomplished by adopting the adaptive group lasso in Wang and Leng (2008) that allowed different amounts of shrinkage on regression coefficients which saw improved variable selection performances over the lasso.

From the literature, we can draw on the strengths of various relevant methods and craft stronger regularization techniques consisting of three main features: i) use the group-norm penalization of individual model parameters to correctly infer null GC from the estimated group-sparse structure, ii) use the non-convex $\ell_{2,1/2}$ penalty to enhance the recovery rate of true parameters, and iii) use relative weights with group penalties to improve the accuracy of detecting edges in an estimated network. All three features, applied to jointly estimate multiple GC networks, form the contributions of this thesis.

We propose three main formulations that each employ all three features listed above: one estimates multiple GC networks to have the same structure, and the other two estimate multiple models inferring both common and individually specific GC networks. An example application of revealing a common GC among multiple models is a group-level inference of brain connectivity, where data sets contain signals of several subjects recorded under a controlled condition (*e.g.*, resting-state), and each model parameter belongs to each patient. Presumably, each patient contributes to a homogeneous brain connectivity structure that can be inferred from the estimated common GC network, while the model parameters are allowed to differ according to each patient's profile. As for examples of discovering differential networks, the brain connectivity structure among subjects would be assumed to contain differences, perhaps arising from the testing of patients under two or more conditions (*e.g.*, control versus abnormal brain patterns). In light of these examples, the effectiveness of our three formulations will be illustrated by using the fMRI ADHD-200 data sets to learn the group-level brain network differences between adolescents with attention deficit hyperactivity disorder (ADHD) and typically developing children (TDC).

Before presenting the main formulations, we provide a background on a single Granger network estimation first in Chapter 2. Then, the estimation formulations and the causality learning scheme, which are the main contribution of this thesis, are presented in Chapter 3. From an algorithmic point of view, we will see in Chapter 4 that our formulations have complications applying the existing algorithms to our non-convex formulations. This is due to a violation of assumptions required in convergence analysis. The convergence issue is addressed by suggesting a heuristic penalty parameter update rule for ADMM to ensure a convergence in practice. For convex formulations, the ADMM algorithm has global convergence for our problems. We also include a penalty parameter update rule for the convex case to speed up the convergence. The effectiveness of the formulations is demonstrated in Chapter 5 through the extensive simulation experiments and through a benchmarking experiment in which we compared our works with the existing literature. The real data experiment was also included in this chapter. Lastly, we concluded the contributions and discuss the possible extension to this thesis in Chapter 6.

1.1 Objectives

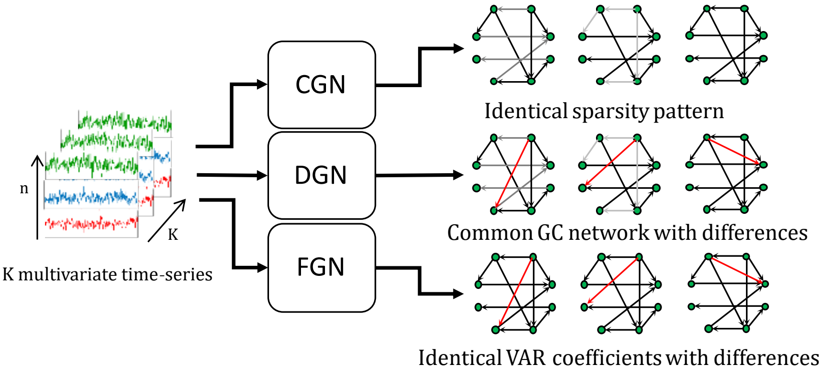

n
K
K multivariate time-series
CGN
DGN
FGN
Identical sparsity pattern
Common GC network with differences
Identical VAR coefficients with differences

Figure 1.1: Characteristics of GC networks learned from the proposed formulations.

1. We aim to propose formulations that take time-series as an input and return VAR models that infer three types of GC networks in the following.

    a) CommonGrangerNet (CGN): The estimated networks have an identical sparsity pattern.

    b) DifferentialGrangerNet (DGN): The estimated networks have some identical parts and some different parts.

    c) FusedGrangerNet (FGN): The estimated networks have some block-identical value of VAR's coefficients and some different sparsity pattern.

    The formulations are presented in Figure 1.1. The dark links are the GC that all models have in common or the common GC network. The red links indicate an individual GC or the differential GC network.

2. We aim to provide efficient numerical methods to solve the proposed estimation methods in a large scale setting.

1.2 Scope of Work

1. The proposed framework will be verified on intensive simulations and one real-world data set.

2. The application of the proposed methods will be illustrated on brain network analysis.

1.3 Expected outcome

1. Estimation formulations of multiple Granger graphical models.

2. Computer program that solve three formulations and perform causality learning.

Chapter II

BACKGROUND

This chapter provides an introduction to the time-series modeling and to the definition of a causality measure called the Granger causality. We also provide a review of a group norm penalty that can induce a block-sparse pattern in the solution, as well as its relation to the Granger graphical modeling.

2.1 Granger causality

The Granger causality (GC) is a measure used to infer causal interconnections among multivariate time-series $\{y(t)\}_{t=1}^{T}$. The computation of GC is often obtained from a p-order vector autoregressive (VAR) model described by

$$y(t) = A_1 y(t-1) + A_2 y(t-2) + \cdots + A_p y(t-p) + \epsilon(t), \tag{2.1}$$

where $y(t) = (y_1(t), y_2(t), \ldots, y_n(t)) \in \mathbf{R}^n$ and $A_r \in \mathbf{R}^{n \times n}$ is the VAR parameters of lag $r = 1, 2, \ldots, p$. The element $(A_r)_{ij}$ indicates a gain from $y_j(t-r)$ to $y_i(t)$. We assume that $\epsilon(t)$ is white Gaussian noise.

An estimation of VAR model's parameters can be cast as a least-square problem,

$$\underset{A}{\text{minimize}} \quad \frac{1}{2N} \|Y - AH\|_F^2, \tag{2.2}$$

where

$$Y = \begin{bmatrix} y(p+1) & y(p+2) & \ldots & y(T) \end{bmatrix} \in \mathbf{R}^{n \times N}, A = \begin{bmatrix} A_1 & \cdots & A_p \end{bmatrix} \in \mathbf{R}^{n \times np}, \tag{2.3}$$

$$H = \begin{bmatrix} y(p) & y(p+1) & \cdots & y(T-1) \\ \vdots & \vdots & \cdots & \vdots \\ y(2) & y(3) & \cdots & y(T-p+1) \\ y(1) & y(2) & \cdots & y(T-p) \end{bmatrix} \in \mathbf{R}^{np \times N}. \tag{2.4}$$

The first p time-points out of T are used as the initial condition for estimation, resulting in $N = T - p$ time-points used in the estimation, called effective time-points. The least-square solution to (2.2) is

$$\hat{A} = YH^T(HH^T)^{-1}, \tag{2.5}$$

which is referred to as *a full model* because it uses all time-series data. The Granger causality from $y_j(t)$ to $y_i(t)$ is based on a comparison between the quality of the full model and the quality of *a reduced model* which is the model that is estimated without a time-series $y_j(t)$ (*i.e.*, least-square fitting to the data $y(t) = (y_1(t), \dots, y_{j-1}(t), y_{j+1}(t), \dots, y_n(t)) \in \mathbf{R}^{n-1}$). If the fitting quality of $y_i(t)$ from two models are the same, it indicates that the existence of $y_j(t)$ is not useful for estimating $y_i(t)$. Thus, the Granger causality from $y_j(t)$ to $y_i(t)$ is defined to be zero.

The quality of the fitting can be quantified using the covariance matrix of the residual of the estimation (*i.e.*, the difference between the time-series and the model's output) or the residual covariance. In general, a smaller residual covariance indicates a higher quality of the fitting. From these concepts, the Granger causality (GC) from $y_j(t)$ to $y_i(t)$ is quantified by the log-det ratio between residual covariance of two cases as

$$\mathcal{F}_{ij} = \log \frac{\det \Sigma_{ii}^R}{\det \Sigma_{ii}}, \tag{2.6}$$

where Σ_{ii}^R is the (i,i) entry of residual covariance of $y_i(t)$ of the reduced model. The term Σ_{ii} is the (i,i) entry of residual covariance of $y_i(t)$ when fitted to a full model. If $\det \Sigma_{ii} < \det \Sigma_{ii}^R$, we obtain $\mathcal{F}_{ij} > 0$ and if $\det \Sigma_{ii} = \det \Sigma_{ii}^R$, we obtain $\mathcal{F}_{ij} = 0$. As a remark, the determinant in (2.6) is necessary since GC is defined in a general case that $y_i(t)$ and $y_j(t)$ can be sub-groups of time-series.

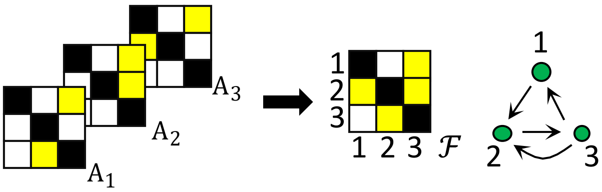


Figure 2.1: Granger causality network and matrix (right) inferred from non-zero index of VAR parameters (left) when $n = 3, p = 3$.

The Granger causality among all time-series formed a Granger graphical model or a GC network. A GC network is a directed graph with asymmetric adjacency matrix as shown in Figure 2.1. The yellow entries indicate the non-zero parameters. Figure 2.1 stressed one of the advantages of Granger graphical model over the Gaussian graphical model that it also provides the directionality of the causal connections.

In practice, a small value of $\mathcal{F}_{ij}$ numerically computed from (2.6) may not be an exact zero. To test the exactness of a zero GC or a null GC, we can either perform a hypothesis test with null hypothesis $\mathcal{F}_{ij} = 0$ against $\mathcal{F}_{ij} \neq 0$ or use the relation,

$$\mathcal{F}_{ij} = 0 \leftrightarrow (A_r)_{ij} = 0, r = 1, 2, \cdots, p, \tag{2.7}$$

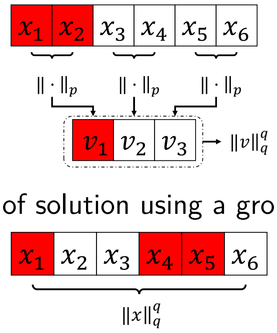
as a prior information for the VAR model estimation. In other words, The VAR coefficients of index (i,j) must be zero in all lags in order to have a null GC from $y_j(t)$ to $y_i(t)$ (Lütkepohl, 2005). We can use a sparse penalty to regularize the objective of (2.2) to force the estimated VAR have sparsity pattern strictly follow the relation (2.7) to obtain a sparse representation of GC network.

2.2 Group norm penalty

Directly from the relation (2.7), the null GC connections can be introduced by using a sparse penalty, g, to convert the problem (2.2) to a regularized least-square problem in the form of

$$\underset{x}{\text{minimize}} \;\; f(x) + \lambda g(x), \tag{2.8}$$

where f is quadratic loss in (2.2). The tuning parameter λ controls the sparsity level of the model. The solution to (2.8) is zero if λ is sufficiently large. One of the sparse penalties is the lasso penalty ($g(x) = \|x\|_1$) (Tibshirani, 1996). However, the lasso penalty is not suitable in many applications because it cannot induced the predefined sparsity pattern. In our case, we required that the sparsity pattern of VAR must follow from (2.7) to effectively obtain a sparse GC network.



(a) The sparsity of solution using a group norm penalty.

(b) The sparsity of solution using a sparse penalty.

Figure 2.2: The difference of sparsity pattern between group norm penalty and a sparse penalty. The **red** variables are the example of possible zero locations.

To induce a sparsity structure, one can regularize the objective in (2.2) using a group norm penalty:

$$g(x; \mathcal{B}) = \sum_{l \in \mathcal{B}} \|x_l\|_p^q, \tag{2.9}$$

which is a composite of ℓ_q with ℓ_p norm. In this case, we desire the sparsity pattern to follow the partition set $\mathcal{B}$. For example in Figure 2.2(a), the partition set $\mathcal{B}$ is $\{\{1,2\},\{3,4\},\{5,6\}\}$ and the possible sparsity patterns are $(x_1, x_2) = 0$ or $(x_3, x_4) = 0$ or $(x_5, x_6) = 0$. The intuition behind this is that the norm of each partition is penalized using a sparse penalty so that the estimated norm of a partition is exactly zero, yielding a zero partition. From Figure 2.2(a), the ℓ_p norm with $p \geq 1$ of each partition forms a vector (v_1, v_2, v_3) which is then penalized using a sparse penalty, ℓ_q with $0 < q \leq 1$, leading to a sparse v. From a definition of a norm, when v_i is zero, the partition of x according to v_i is also a zero vector, producing a block-sparse solution in the original variable, x. Without the group norm penalty, the sparsity of the solution may not exactly follows from the predefined pattern as shown in Figure 2.2(b). The usage of conventional sparse penalty in the problem that the sparsity structure is meaningful can introduce the extra bias from unnecessary zero variables in the estimation without learning a desired sparsity structure.

Chapter III

METHODOLOGY

This chapter is concerned with the main contribution of the thesis. The basic information on the joint sparse Granger graphical model estimation is given in the first part. The description of our proposed regularization techniques are provided in the second part. The last part of this chapter, the GC learning scheme, focus on the applications of the proposed techniques.

Notation A vector x is partitioned into m blocks as $x = (x_1, x_2, \dots, x_m)$ and a group-norm $\ell_{p,q}$ of x is defined as $\|x\|_{p,q} = (\sum_i \|x_i\|_p^q)^{1/q}$. We often use the $\ell_{p,q}$ norm to the power of q. $\|\cdot\|_p$ indicates p norm.

A joint Granger graphical model is a collection of K p^{th} VAR models,

$$y^{(k)}(t) = \sum_{r=1}^{p} A_r^{(k)} y^{(k)}(t-r) + \epsilon^{(k)}(t), \quad k = 1, \dots, K, \tag{3.1}$$

with predefined relations among all models as a prior knowledge. The prior knowledge takes the form of a regularization of objective in (2.2) of K models as

$$\underset{A^{(1)},\dots,A^{(K)}}{\text{minimize}} \; \frac{1}{2N} \sum_{k=1}^{K} \|Y^{(k)} - A^{(k)} H^{(k)}\|_F^2 + g(A^{(1)}, \dots, A^{(K)}), \tag{3.2}$$

where $Y^{(k)}, H^{(k)}$ are the matrices containing measurements of k^{th} time-series and $A^{(k)}$ are VAR coefficients of k^{th} model. The expression of $Y^{(k)}, H^{(k)}$ and $A^{(k)}$ is provided in (2.3), (2.4).

Without prior knowledge of the relations among all models, the estimation in (3.2) is reduced to K separable optimization problems for each $A^{(k)}$ and can be solved in parallel. In such case, we can learn K sparse GC networks by using a weighted group norm penalty function,

$$g(A) = \lambda \sum_{k=1}^{K} \sum_{i \neq j} w_{ij}^{(k)} \|B_{ij}^{(k)}\|_2^q, \quad 0 < q \leq 1, \tag{3.3}$$

where,

$$B_{ij}^{(k)} = \begin{bmatrix} (A_1^{(k)})_{ij} & \cdots & (A_p^{(k)})_{ij} \end{bmatrix} \in \mathbf{R}^p \tag{3.4}$$

This choice of penalty extends the joint Granger graphical model estimation framework in three ways.

- First, this penalty directly follows from Chapter 2 that it penalizes the partition $B_{ij}^{(k)}$ to be zero, leading to a null GC for k^{th} model. We emphasize on the use of $\|B_{ij}^{(k)}\|_2^q$ instead of using the conventional lasso of VAR coefficients directly as $\sum_{r=1}^{p} |(A_r^{(k)})_{ij}|$ (lasso penalty) as it does not promote a *common zero* among *all VAR-lag* parameters.

- Second, we include the choice of penalty weight $w_{ij}^{(k)}$ in (3.3) since a pre-defined $w_{ij}^{(k)} > 0$ indicates the degree to which each $B_{ij}^{(k)}$ is penalized, or equivalently, it gives a likelihood prior of GC from $y_j^{(k)}$ to $y_i^{(k)}$. The topology of GC network of the k^{th} model refers to the sparsity pattern of an $n \times n$ matrix formed by the estimated $B_{ij}^{(k)}$ for $1 \leq i, j \leq n$.

- Third, we extends to use the non-convex group norm penalty. When $p = 2, q < 1$, the problem is reduced to the non-convex group norm regularized regression which was studied in the context of group sparsity recovery in Hu et al. (2017). A recovery bound of group-sparse solutions to an $\ell_{p,q}$-regularized regression can be guaranteed upon the (p, q)-group restricted eigenvalue condition (GREC) Hu et al. (2017)[1] with an important property that (2,1)-GREC implies (2,1/2)-GREC. This favorable result implied that using the $\ell_{2,1/2}$ penalty requires a weaker condition than $\ell_{2,1}$, that is called the adaptive group lasso Wang and Leng (2008), to obtain the recovery bound. The experimental results of Hu et al. (2017) suggested that the range of true sparsity levels that yielded 100% success recovery rate in $\ell_{2,1/2}$ was wider than that of $\ell_{2,1}$.

The penalty (3.3) is yet to be a joint modeling framework but can be converted into one by modifying the structure of group norm penalty to match three assumptions of the relation among models presented in Figure 3.1. The **red** links indicate the differential GC link, the **dark** links indicate the common GC link and the color intensity indicates GC strength. In summary, the assumptions are

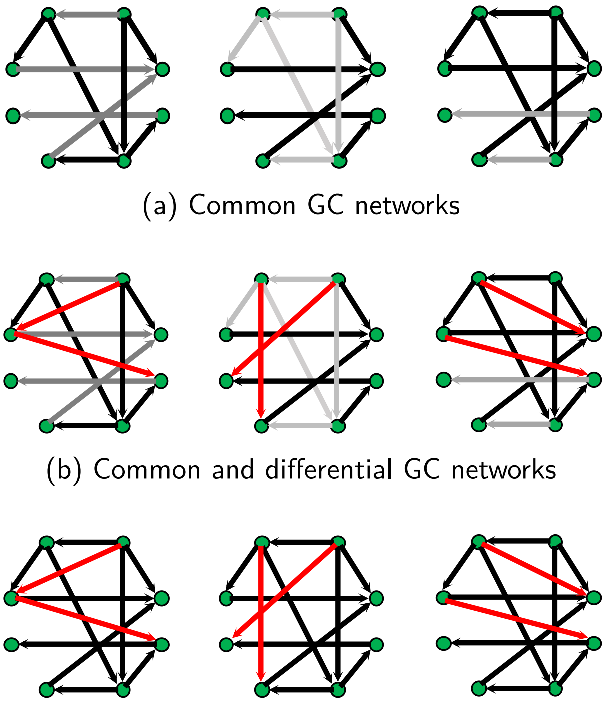
1. Common GC networks assumption: all models shared the GC connections without sharing the value. This common topology is called the **common GC network**,

2. Common and differential GC networks assumption: all models partially shared their topology and each model also has GC connections different from the common part called the **differential network**,

3. Fused and differential GC networks assumption: the definition is the same as assumption 2 but the common GC network in each model has same VAR coefficients.

Their expressions and related literature are presented in the following sections.

[1] The condition requires the positive definiteness of $A^T A$ on the associated subblocks. The bound of the estimation error is a big $\mathcal{O}$ of λ and the group sparsity level of the true parameter; see Theorem 9 in Hu et al. (2017).



(a) Common GC networks

(b) Common and differential GC networks

(c) Fused and differential GC networks

Figure 3.1: The assumptions on each joint Granger graphical model estimation when $K = 3$.

3.1 Common GC network

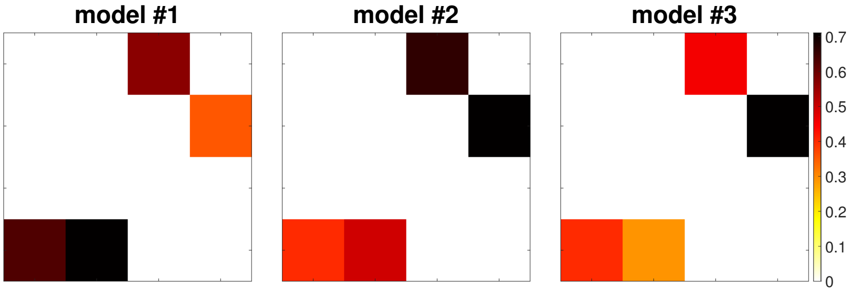


Figure 3.2: Example of GC networks learned with **CGN** formulation.

We propose **CommonGrangerNet (CGN)** as the formulation for estimating K models to have the same GC network topology but their value can be different as shown in Figure 3.2 for the case $K = 3$. A common sparsity of GC network can be obtained by pooling $B_{ij}^{(k)}$ from K models into

$$C_{ij} = \begin{bmatrix} B_{ij}^{(1)} & B_{ij}^{(2)} & \cdots & B_{ij}^{(K)} \end{bmatrix} \in \mathbf{R}^{pK}, \tag{3.5}$$

and regularizing C_{ij} using the group norm penalty

$$g(A) = \lambda \sum_{i \neq j} v_{ij} \|C_{ij}\|_2^q, \quad 0 < q \leq 1. \tag{3.6}$$

As the summation over (i, j) of non-negative quantities behaves like an ℓ_1 penalty, when the penalty parameter λ is sufficiently large, some C_{ij}'s (from $1 \leq i, j, \leq n$) are zero, revealing a common GC structure of K models. Thanks to the characteristics of group norm penalty, the value of non-zero $B_{ij}^{(k)}$ does not require to be the same for all k. The relative penalty weight among (i, j) is chosen as $v_{ij} = 1/\|\tilde{C}_{ij}\|_2^q$ where $\tilde{C}_{ij}$ is the least-squares estimate of C_{ij}. This choice is suggested from that if a norm of the $\tilde{C}_{ij}$ is very small compared to others, it is more likely that $C_{ij} = 0$; thus, C_{ij} should be more penalized and vice versa. When $q = 1$, this technique is also known as adaptive group lasso (Wang and Leng, 2008). The penalty (3.6) was considered in Songsiri (2017); Gregorova et al. (2015) but with $q = 1, v_{ij} = 1$.

3.2 Common and differential GC network

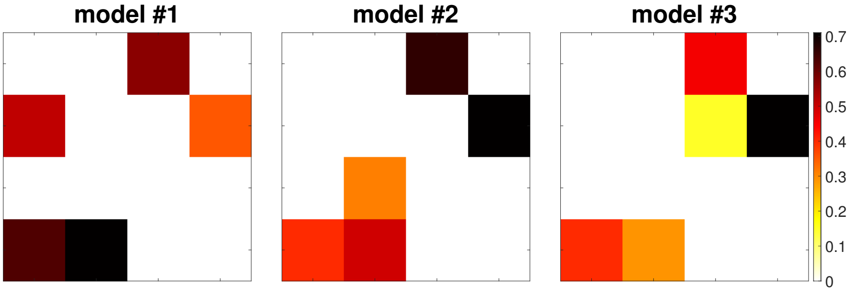


Figure 3.3: Example of GC networks learned with **DGN** formulation.

In the case that the differences of individual models are favored to learn, CGN cannot be applied. The differences of GC networks, called the differential GC networks, can be introduced by combining the term (3.3) to (3.6) as

$$g(A) = \lambda_1 \sum_{i \neq j} \sum_{k=1}^{K} w_{ij}^{(k)} \|B_{ij}^{(k)}\|_2^q + \lambda_2 \sum_{i \neq j} v_{ij} \|C_{ij}\|_2^q, \quad 0 < q \leq 1. \tag{3.7}$$

The second term in (3.7) can promote a *shared* null GC in some (i, j) entries by *all* GC networks, while other (i, j) entries of *individual* models can be regularized through the first term of g. With the penalty in (3.7), the models are encouraged to decompose into the common networks and the differential networks as shown in Figure 3.3. Similar to (3.6), the relative weights are chosen as the inverse of the least-square estimate: $w_{ij}^{(k)} = 1/\|\tilde{B}_{ij}^{(k)}\|_2^q$ and $v_{ij} = 1/\|\tilde{C}_{ij}\|_2^q$. The estimation (3.2) with penalty (3.7) is referred to as **Differential-GrangerNet (DGN)**.

Previous works on the regularization techniques that split the GC networks into common and differential parts include Songsiri (2017); Skripnikov and Michailidis (2019b). Songsiri (2017) proposed the penalty term that is the special case of (3.7) with $q = 1, w_{ij}^{(k)} = 1, v_{ij} = 1$. In other words, they used convex penalties (group lasso) and did not provide the choices of the relative weight on model parameters. Skripnikov and Michailidis (2019b) proposed a different technique for common-differential network decomposition. Their method consisted of two stages, they learned the common part in the first stage and the differential part in the second stage. Despite the convex penalty they used, their two-stages formulation is non-convex. The global optimality of their formulation cannot be ensured.

3.3 Fused and differential GC network

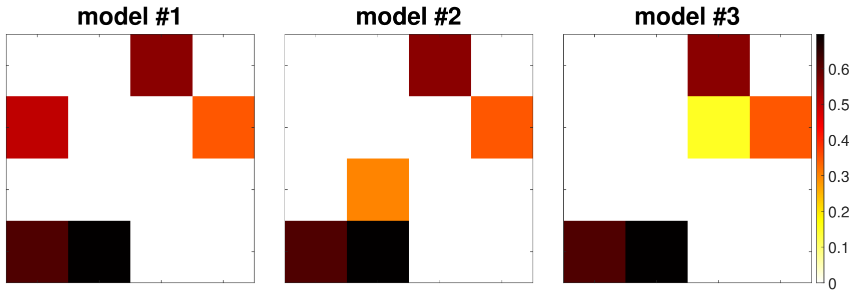


Figure 3.4: Example of GC networks learned with **FGN** formulation.

In addition to the DGN formulation, we assume that the common Granger networks share the same VAR coefficients, $B_{ij}^{(1)} = B_{ij}^{(2)} = \cdots = B_{ij}^{(K)}$ as shown in Figure 3.4. The common value of GC can be obtained by replacing the second term in (3.7) to be a group fused lasso penalty (Alaíz et al., 2013) as

$$g(A) = \lambda_1 \sum_{i \neq j} \sum_{k=1}^{K} w_{ij}^{(k)} \|B_{ij}^{(k)}\|_2^q + \lambda_2 \sum_{k<l} \sum_{i \neq j} u_{ijkl} \|B_{ij}^{(k)} - B_{ij}^{(l)}\|_2^q, \quad 0 < q \leq 1, \tag{3.8}$$

that shrink the differences between $B_{ij}^{(k)}$ and $B_{ij}^{(l)}$ in all combination of l, k through the sum $\sum_{k<l}$. A common GC of two models is introduced when their difference is shrunk to zero for some (i,j) as shown in Figure 3.4. The positive penalty weight u_{ijkl} gives a relative degree of penalizing the group fused lasso among (i,j) entries and all pairs of model k and l. We can select the penalty weight as, $u_{ijkl} = 1/\|\tilde{B}_{ij}^{(k)} - \tilde{B}_{ij}^{(l)}\|_2^q$ where $\tilde{B}_{ij}^{(k)}$ is the least-square estimate because when $\tilde{B}_{ij}^{(k)}, \tilde{B}_{ij}^{(l)}$ are close to each other, the two vectors are likely to be equal so the larger penalty weight is favored. We refer to this formulation as **FusedGrangerNet (FGN)**.

Relevant literature of this formulation includes Skripnikov and Michailidis (2019a); Songsiri (2015, 2017). In Skripnikov and Michailidis (2019a), each vector in the fused term was constructed differently from (3.8); it was pooled from *all entries* of VAR-lag coefficients of a single model, while in FGN, it was pooled from an (i,j) entry of lag coefficients. The two techniques can force identical parameters of two models due to the fused-lasso feature. However, when $p > 1$, we desire fusion results that the sparsity of single model's parameters occur as a *group* of all VAR-lag coefficients for some (i,j); such zero patterns can characteristically infer causality from variable j to i, which can be achieved by (3.8), but not Skripnikov and Michailidis (2019a). Despite penalizing all of any two models in (3.8), the fused terms in Songsiri (2015, 2017) penalized only two consecutive models and did not apply prior information about the common and differential part by simply using $u_{ijkl} = 1, w_{ij}^{(k)} = 1$, while our choice of $u_{ijkl}, w_{ij}^{(k)}$ can suggest at which (i, j) entry is zero and which pair of two models have more likelihood of having identical parameters.

3.4 GC learning scheme

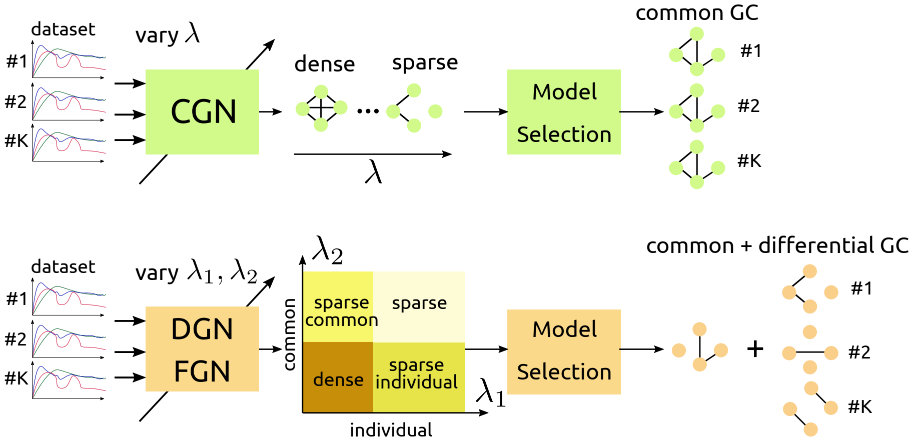


Figure 3.5: Learning scheme for the proposed formulations.

This section is primarily focused on the applications of the proposed formulations which is summarized in Figure 3.5. The process of learning the common & differential GC networks begins with feeding the K time-series into the desired formulation. To find a candidate for inference, the candidates are obtained by varying the tuning parameters that are used in the formulation and the candidate is selected by a model selection technique or the penalty parameter selection. The selected model is then decomposed into common and differential networks which can be used for the inference task. In the following, we provides the detail of each learning step.

Penalty parameter selection

Selecting a suitable penalty parameter is to seek an optimal trade-off between model complexity and the fitting of the model using a model selection technique. To select a model, we propose to use eBIC (Extended Bayesian information criterion) (Chen and Chen, 2008),

$$\text{eBIC}(\lambda_1, \lambda_2) = -2\mathcal{L}(\lambda_1, \lambda_2) + \text{df}(\lambda_1, \lambda_2) \log(N) + 2\gamma \log\left(\frac{n^2 pK}{\text{df}(\lambda_1, \lambda_2)}\right); \quad 0 \leq \gamma \leq 1, \tag{3.9}$$

where $\mathcal{L}(\lambda_1, \lambda_2)$ is the log-likelihood of the model obtained from the model estimated with λ_1, λ_2; $N = T - p$ is number of effective time points; $\text{df}(\lambda_1, \lambda_2)$ is the degrees of freedom (or the complexity measure of a model). We assume that $\epsilon^{(k)}(t)$ in (3.1) are independent in each k so that the log-likelihood of K VAR models is,

$$\mathcal{L}(\lambda_1,\lambda_2) = -\frac{nNK}{2}\log(2\pi) - \frac{N}{2}\sum_{k=1}^{K}\log\det\hat{\Sigma}^{(k)}(\lambda_1,\lambda_2) - \frac{nNK}{2}$$

where $\hat{\Sigma}^{(k)}(\lambda_1,\lambda_2)$ is the maximum likelihood estimate of k^{th} model's noise covariance. We provide the derivation of the log-likelihood in Appendix B.1. The term $\binom{n^2pK}{\text{df}}$ in (3.9) represents the number of possible model candidates with a given df. This term represents the log-prior distribution of model that varies upon model's df. The amount of prior information can be adjusted through a tuning parameter (γ) that eventually affects eBIC asymptotic property. When $\gamma = 0$, the eBIC expression is reduced to BIC score. The expression of df also affects the asymptotic properties of the criteria (Hastie et al., 2001) and the suitable choice of df is varied from problem to problem. For a lasso regression, the degrees of freedom are simply the non-zero estimated variables. For a fused-lasso problem, the multiple variables sharing the same value are regarded as one degree of freedom. For a group lasso problem, the degrees of freedom are approximated from the number of non-zero groups and the ratio between regularized solution norm and least-square solution norm of a group (Yuan and Lin, 2006). Since CGN is essentially an extension to the group lasso regression, we follow the choice of degrees of freedom from Yuan and Lin (2006):

$$\text{df}(\lambda) = \sum_{i,j}\mathcal{I}(\|C_{ij}(\lambda)\|_2 > 0) + \sum_{i,j}\frac{\|C_{ij}(\lambda)\|_2}{\|\tilde{C}_{ij}\|_2}(pK-1),$$

where $C_{ij}(\lambda)$ and $\tilde{C}_{ij}$ are the solution of CGN and the least-square solution respectively. The notation $\mathcal{I}(\mathcal{S})$ denotes the indicator function that return 1 if the statement $\mathcal{S}$ holds and return 0 otherwise. This choice of degrees of freedom cannot be applied in other formulations due to a different group size in DGN and a different type of penalty in FGN (group fused lasso). We heuristically count the non-zero variables in DGN and counted the fused variables (*i.e.* the variables that shared their value) as one degree of freedom in FGN. We select a pair of penalty (λ_1,λ_2) if it minimizes (3.9) or

$$(\hat{\lambda}_1,\hat{\lambda}_2)_{\text{eBIC}} = \underset{\lambda_1,\lambda_2}{\text{argmin}}\ \text{eBIC}(\lambda_1,\lambda_2).$$

As a remark, there are alternative approaches of choosing the penalty parameters other than the model selection criterion. We consider **K-fold cross validation (CV)** and **stability selection**. The goal of K-fold CV is to seek a model that has the lowest averaged sum-square error (SSE) of unseen data over K sub-samples. As the name suggested, the K-fold CV splits the data into K non-overlapping sub-samples. One out of K segment is selected to be a validation data (or the unseen data), y_{validate}, and the rest is called the training data y_{train} which is used in the model estimation. The model with a given regularization is then estimated using y_{train} and evaluating using y_{validate}, yielding a sum-square error loss $\text{SSE}^{(k)}$ where k is the validation data segment index. The iteration is then repeated for K time. In summary, the sum-square error for all segments are evaluated as

$$\mathsf{SSE}^{(k)}(\lambda_1,\lambda_2) = \|(y^{(k)}_{\mathsf{validate}} - \hat{y}^{(k)}_{\mathsf{validate}}(\lambda_1,\lambda_2))\|_2^2, \quad k = 1,\ldots,K,$$

where $y^{(k)}_{\mathsf{validate}}$ is the k^{th} validation data segment and $\hat{y}^{(k)}_{\mathsf{validate}}(\lambda_1,\lambda_2)$ is the output of the model estimated with penalty (λ_1,λ_2) using training data. The process repeated by iterating through all K segments and repeated over all regularization pairs (λ_1,λ_2). The result of K-fold CV is the averaged sum-square-error over K-fold as

$$\mathsf{SSE}_{\mathsf{avg}}(\lambda_1,\lambda_2) = (1/K)\sum_{k=1}^{K}\mathsf{SSE}^{(k)}(\lambda_1,\lambda_2).$$

The CV method selects the regularization pair such that $\mathsf{SSE}_{\mathsf{avg}}$ is minimized or

$$(\hat{\lambda}_1,\hat{\lambda}_2)_{\mathsf{CV}} = \underset{\lambda_1,\lambda_2}{\mathsf{argmin}}\ \mathsf{SSE}_{\mathsf{avg}}(\lambda_1,\lambda_2).$$

However, the goal of K-fold CV is to minimize the prediction error in the unseen data set which is a different objective for the true model selection. In practice, the K-fold CV is prone to select a dense model.

The stability selection (Meinshausen and Bühlmann, 2010) focuses on the zero location consistency of the estimated parameters when the data were sub-sampled. Their concept is that when the data were sub-sampled, the zero index of the estimated parameters should not be significantly affected. The stability selection begins with randomly drawing half of the samples as a sub-sample. Each sub-sample is used as data of the sparse estimation and the process is repeated for K times with the replacement of the drawn data for each penalty parameter. For a given tuning parameter, the ratio of the times that a variable is estimated as non-zero over K times is assigned to each variable and is referred to as a stability measure of an estimated variable. The variables with higher stability measure than a given threshold are regarded as the stable variables. Unlike the other model selection techniques, the stability selection is not a direct way for selecting (λ_1,λ_2). It is rather an ensemble method that is used to select stable variables over a given range of regularization.

These two techniques have two main drawbacks in the time-series estimation context. First, they required solving the estimation for a given penalty repeatedly. This is not feasible for a high dimensional setting. This repetition is not required in the eBIC selection. Second, the time-series samples are highly time-dependent and cannot be randomly sub-sampled. One can sub-sample the time-series as a block of smaller time-series but the estimation with smaller samples may result in poor goodness of fit. Based on these reasons, we prefer the eBIC selection over the sub-sampling-based model selection.

Critical penalty parameter

Since any sufficiently large penalty parameters will yield a sparsest model, the bound of penalty grid search should be set by the smallest penalty parameter called the critical penalty parameter λ_c. If the bound is tight, the estimated models are not entirely sparse for all $\lambda < \lambda_c$ but entirely sparse for $\lambda \geq \lambda_c$. After the bound is determined, we can freely set the resolution of the grid search as we desired over $0 < \lambda \leq \lambda_c$. The expression of λ_c for cvx-CGN has been given in Songsiri (2017). The following is its expression based on the vectorized form of (4.1) which we will revisit in this topic in Chapter 4.

$$\lambda_c = \max_{i \in J} \; (1/w_i) \| G_i^T [I - G_q (G_q^T G_q)^{-1} G_q^T] b) \|_2, \tag{3.10}$$

where G_q is the block-column of G corresponding to the unpenalized parameters (diagonal term of VAR models) and G_i is the block-column of G that associated with the i^{th} block of off-diagonal VAR coefficients respectively. The set J contains indices of the penalized blocks with penalty weight w_i.

We denotes $(\lambda_{1c}, \lambda_{2c})$ to be the critical penalty parameters for DGN and FGN. For these formulations, we heuristically compute $\lambda_{1c}, \lambda_{2c}$ by determining the analytical form of λ_{2c} based on $\lambda_1 = 0$ and λ_{1c} based on $\lambda_2 = 0$. In DGN, we heuristically computed λ_c for λ_1, called λ_{1c}, in the case that λ_2 is zero and the same for λ_{2c}. In FGN, we also set λ_{1c} the same way as DGN but we heuristically set $\lambda_{2c} = \lambda_{1c}$. When the formulations are non-convex, the bound (3.10) is no longer tight; however, the bound is still usable because a non-convex formulation yields a sparser solution than the convex formulation.

Extracting common and differential network

The K estimated GC networks can be decomposed into two parts, the common GC network and differential GC networks. For the k^{th} network, we define its edge list, or a set containing all of non-zero edges as $F^{(k)}$. The edge list of common GC network, F_{common}, is defined from the intersection of all edge lists for all K as

$$F_{\text{common}} = \bigcap_{k=1}^{K} F^{(k)}$$

forming a common GC network. Unlike FGN, the common GC links of CGN and DGN may not share the same weight, leading to ambiguous choices of weight. We heuristically define the weight of common GC links to be the average of all common GC networks. For a k^{th} differential network, its edge list, $F^{(k)}_{\text{differential}}$, is defined from the edges of the k^{th} GC network that is not in F_{common} or,

$$F^{(k)}_{\text{differential}} = F^{(k)} \setminus F_{\text{common}} \quad k = 1, \dots, K.$$

The goals of causality learning based on common network and differential networks is divided into two objectives: focusing on F_{common} and focusing on $F^{(k)}_{\text{differential}}$ for $k = 1, \dots, K$. For the first goal, while each model may contain a different intrinsic GC structure, a meaningful group-level characteristic is preferred. CGN formulation can directly serves this purpose since

$F^{(k)}_{\text{differential}} = \emptyset$ for all k. For DGN and FGN, the edges list, F_{common}, can be extracted from the overlapped non-zeros of $B^{(k)}_{ij}$ among k. FGN is suitable for this objective since the coefficients in the estimated common part already share their value for all K models. A wide application of revealing a common GC is a group-level inference of brain connectivity where data sets contain brain signals of several subjects collected under a controlled condition (*e.g.*, resting-state), and K is then the number of patients. Presumably, these subjects contribute homogeneous brain connectivity that can be inferred from the estimated GC common network, while the model parameters are allowed to differ for each patient's profile. The second goal aims at investigating the differences among multiple networks directly. After the common part is specified, the differential network of each model corresponds to the remaining non-zero locations of $B^{(k)}_{ij}$ for each k. As examples of discovering differential networks, we aim to infer brain connectivity differences from the brain signals collected under two or more conditions (*e.g.*, controlled versus abnormal). The second goal can be served only from DGN and FGN; however, for FGN, the fused term regularized the differential networks, causing the differences to be small. Therefore, DGN may be more suitable for the second objective.

Chapter IV

ALGORITHMS

For ease of mathematical presentation, we reformulate the problem formulations CGN, DGN, and FGN in a vector format (without scaling N) as

$$\underset{x}{\text{minimize}} \quad (1/2)\|Gx-b\|_2^2 + g(x), \tag{4.1}$$

where the optimization variable $x \in \mathbf{R}^{n^2pK}$ refers to the n-dimensional p-order VAR model parameters of K models: $A^{(k)}$ for $k = 1, \dots, K$. The problem parameters, $b \in \mathbf{R}^{nNK}, G \in \mathbf{R}^{nNK \times n^2pK}$, are vectorized from $Y^{(k)}$ and $H^{(k)}$ for all K respectively. The first term in the objective of (4.1) represents the least-squares objective (model fitting) and the second term, $g(x)$, is the regularization funtion that depends on the formulation. For all formulations, the penalty term is either based on $B_{ij}^{(k)}$ in (3.4) or C_{ij} in (3.5). Therefore, it is more convenient to present the variable x in the form of

$$x = (C_{11}, \dots, C_{1n}, C_{21}, \dots, C_{2n}, \dots, C_{n1}, \dots, C_{nn}) \in \mathbf{R}^{n^2pK}, \tag{4.2}$$

so that vector x can be partitioned either into $B_{ij}^{(k)}$ as a sub-block of size p or into C_{ij} as a sub-block of size pK. Based on different size of sub-block, we define a partition set containing index according to sub-block $B_{ij}^{(k)}$ as

$$\begin{aligned}\mathcal{P} = \{&\{1,2,\dots,p\}, \{p+1, p+2, \dots, 2p\}, \dots, \\ &\{(B_1-1)p+1, (B_1-1)p+2, \dots, B_1p\}\},\end{aligned} \tag{4.3}$$

where B_1 is the number of all partitions and according to C_{ij} as

$$\begin{aligned}\mathcal{K} = \{&\{1,2,\dots,pK\}, \{pK+1, pK+2, \dots, 2pK\}, \dots, \\ &\{(B_2-1)pK+1, (B_2-1)pK+2, \dots, B_2pK\}\},\end{aligned} \tag{4.4}$$

with B_2 as the number of all partitions. With the notation of partitions, we can adopt group norm penalty from (2.9) with a penalty weight $w_l > 0$ as

$$h(x; \mathcal{B}) = \sum_{l \in \mathcal{B}} w_l \|x_l\|_2^q. \tag{4.5}$$

The penalization of the partitions in $\mathcal{B} = \mathcal{K}$ introduces a common Granger network. Similarly, the penalization of the partition in $\mathcal{B} = \mathcal{P}$ introduces differential Granger networks.

Because the meaningful GC is based on the off-diagonal parts of VAR coefficients, we penalize the projected coordinate, Px, where $P \in \mathbf{R}^{(n^2-n)pK \times n^2pK}$ is a projection mapping that maps VAR coefficients, x, onto the off-diagonal parts of VAR coefficients. To show the structure of P; consider the case when $n = 2$, we have $x = (C_{11}, C_{12}, C_{21}, C_{22})$ with $C_{ij} \in \mathbf{R}^{pK}$. The projection matrix P is given by

$$P = \begin{bmatrix} 0 & I_{pK} & 0 & 0 \\ 0 & 0 & I_{pK} & 0 \end{bmatrix},$$

so that the projected vector, $Px = (C_{12}, C_{21})$, contains only off-diagonal entries of VAR coefficients of K models. Therefore, the group norm penalty (4.5) of the off-diagonal VAR coefficients is $h(Px; \mathcal{B})$.

Similarly for FGN, we are interested in the regularization on the differences of off-diagonal parameters among all models. For a vectorized variable x, we can define the difference operator $D \in \mathbf{R}^{(n^2-n)p\binom{K}{2} \times n^2pK}$ that maps all VAR coefficients to the difference of off-diagonal VAR coefficients between any two models. To show a structure of D; consider the case when $K = 3$ and for a fixed (i, j), the term Dx can be constructed by concatenating

$$\begin{bmatrix} B_{ij}^{(1)} - B_{ij}^{(2)} \\ B_{ij}^{(1)} - B_{ij}^{(3)} \\ B_{ij}^{(2)} - B_{ij}^{(3)} \end{bmatrix} = \begin{bmatrix} I_p & -I_p & 0 \\ I_p & 0 & -I_p \\ 0 & I_p & -I_p \end{bmatrix} \begin{bmatrix} B_{ij}^{(1)} \\ B_{ij}^{(2)} \\ B_{ij}^{(3)} \end{bmatrix} \triangleq \left(\underbrace{\begin{bmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix}}_{D_K} \otimes I_p \right) \begin{bmatrix} B_{ij}^{(1)} \\ B_{ij}^{(2)} \\ B_{ij}^{(3)} \end{bmatrix}, \tag{4.6}$$

for all $1 \leq i, j \leq n$ and $i \neq j$. For a general K, we see that D_K takes all possible differences between any two entries, so the structure of D_K depends on K and its dimension is $\binom{K}{2} \times K$. Define $\tilde{D}$ as the $(n^2 - n)$-block diagonal matrix with all blocks of $D_K \otimes I_p$. Suppose $z = Px$ then z contains only off-diagonal entries of VAR parameters. If z is partitioned to blocks of size p, *i.e.*, each block is $B_{ij}^{(k)}$ as in (4.6), then we see that $\tilde{D}z$ is Dx as desired. Mathematically, $D = \tilde{D}P$ but this expression should not be used in the numerical construction of Dx. Therefore, the weighted group norm penalty for FGN is $h(Dx; \mathcal{B})$.

The vectorized form of g in all formulations are

$$\text{CGN}: g(x) = \lambda h(Px; \mathcal{K}), \tag{4.7}$$

$$\text{DGN}: g(x) = \lambda_1 h(Px; \mathcal{P}) + \lambda_2 h(Px; \mathcal{K}), \tag{4.8}$$

$$\text{FGN}: g(x) = \lambda_1 h(Px; \mathcal{P}) + \lambda_2 h(Dx; \mathcal{P}). \tag{4.9}$$

Therefore, a unified formulation for all of our vector formulations is

$$\underset{x}{\text{minimize}} \quad (1/2)\|Gx - b\|_2^2 + \lambda_1 h(L_1 x; \mathcal{B}_1) + \lambda_2 h(L_2 x; \mathcal{B}_2). \tag{4.10}$$

We provided the vectorization detail in Appendix B.2.

Because the gradient at zero of the objective function (4.10) is undefined, the notion of non-smooth optimization must be introduced. The first part of this section introduces

Table 4.1: The convergence property and convergence rate for CGN, DGN, FGN formulations and their convex relaxations. The symbol ✓ indicates the algorithm in the row has a global convergence and can be efficiently implemented to solve the formulation in the column. The symbol (✓) indicates a global convergence but without efficient implementation.

(a) Algorithm convergence and rate for **convex** formulations.

Available algorithms	cvx-CGN		cvx-DGN		cvx-FGN	
	Convergence	rate	Convergence	rate	Convergence	rate
nmAPG (Li and Lin, 2015)	✓	$\mathcal{O}(1/k^2)$	(✓)	$\mathcal{O}(1/k^2)$	(✓)	$\mathcal{O}(1/k^2)$
Spectral ADMM (Xu et al., 2017b)	✓		✓		✓	
Adaptive ADMM	✓	$\mathcal{O}(1/k)$	✓	$\mathcal{O}(1/k)$	✓	$\mathcal{O}(1/k)$

(b) Algorithm convergence for **non-convex** formulations.

Available algorithms	Convergence		
	CGN	DGN	FGN
nmAPG (Li and Lin, 2015)	✓	(✓)	(✓)
Spectral ADMM (Xu et al., 2017b)			
Adaptive ADMM			

(c) Algorithm complexity per iteration based on the number of model parameters $\tilde{n} = n^2pK$.

Available algorithms	Worst case computation complexity per iteration
nmAPG	$\mathcal{O}(\tilde{n}^2)$ (Matrix-vector multiplication)
Spectral ADMM	$\mathcal{O}(\tilde{n}^3)$ (Solving system of linear equations)
Adaptive ADMM	$\mathcal{O}(\tilde{n}^3)$ (Solving system of linear equations)

readers to a class of algorithms called the proximal algorithms which are mainly used in non-smooth convex optimization problems. Since some of our formulations are non-smooth non-convex problems, we divide the contents into two major parts: the algorithms for convex and non-convex formulations. In brief, we provide a list of algorithms that we used in our implementation in Table 4.1(a) for convex formulations and Table 4.1(b) for non-convex formulations. The known convergence result and the convergence rate is also presented. The notation k is the iteration index, and let $\epsilon = f(x^{(k)}) - p^*$ be the accuracy of objective compared to the optimal value p^*. The convergence rate is reported as a big $\mathcal{O}$ of functions in k which indicates the accuracy ϵ as a function of k. An algorithm with lower accuracy at the same number of iterations compared to another algorithm is faster. For example, $\mathcal{O}(1/k^2)$ has a faster convergence rate than $\mathcal{O}(1/k)$. The worst case computation complexity per iteration and the description is presented in Table 4.1(c).

4.1 Proximal algorithms

This section provides a brief summary of the proximal algorithms in Parikh and Boyd (2014). The algorithm in this thesis often involves a proximal operator of a function h defined by

$$\mathbf{prox}_{\lambda h}(v) = \underset{x}{\operatorname{argmin}}\ h(x) + \frac{1}{2\lambda}\|x - v\|^2, \tag{4.11}$$

where $x, v \in \mathbf{R}^n, \lambda > 0$. The proximal operator (4.11) is usually presented as a sub-problem in a class of many algorithms called the proximal algorithms. It is worth noting that many proximal algorithms do not require the objective to be globally differentiable, so they are more favorable in the non-smooth optimization problems. When compared to the gradient methods, the proximal operator is not intuitive; however, it can be interpreted in many ways. The most natural way is the interpretation as a gradient flow system. Consider the system

$$\dot{x}(t) = -\nabla f(x(t)), \tag{4.12}$$

where a stationary point of the system is the local optima of f. By using Forward-Euler discretization with step size λ; $\dot{x}(t)$ is discretized to $(x(t+\lambda) - x(t))/\lambda$. For notation simplicity, we refer $x(t)$ as x_k and $x(t+\lambda)$ as x_{k+1} . We then obtain the discretized gradient flow system,

$$\frac{x_{k+1} - x_k}{\lambda} = -\nabla f(x_k), \tag{4.13}$$

which is the gradient descent algorithm in disguise. Similarly for Backward-Euler discretization, we obtain

$$\frac{x_{k+1} - x_k}{\lambda} = -\nabla f(x_{k+1}). \tag{4.14}$$

It can be seen that (4.14) is the first order optimality condition of the problem (4.11) when $v = x_k$. Therefore, the Backward-Euler discretized gradient flow eqrefeq:Backward can be replaced by a fixed-point iteration,

$$x_{k+1} = \mathbf{prox}_{\lambda f}(x_k),$$

which is known as the *proximal point* algorithm. Like the gradient descent, a closed-form expression of the proximal operator reduces the computation cost. However, many problems, such as regularized regression problems, are in the form of

$$\underset{x}{\text{minimize}}\ f(x) + g(x),$$

where term f is the smooth fitting term such as the sum-square loss and g is a non-smooth convex regularization such as the ℓ_1 norm penalty. Even when a closed-form of $\mathbf{prox}_g$ is known, $\mathbf{prox}_{f+g}$ is unnecessarily obtained in a closed-form for a general f; therefore, the proximal point algorithm cannot be directly applied. To analyze further, we require a concept of the subdifferentials since the term g is non-smooth.

With a presence of gradient discontinuity for a convex function, the notion of subgradients must be introduced. For a convex f, the set of all vectors s that satisfied the inequality:

$$f(x) \geq f(z) + s^T(x - z), \quad \forall x, z \in \mathbf{R}^n,$$

are called the subdifferential of $f(x)$ at $x = z$ and its members are called subgradients. Geometrically, a subgradient s is any supporting hyperplane of $f(x)$ at $x = z$. The subdifferential of $f(x)$ at $x = z$ is denoted as $\partial f(z)$. As an example, the subgradient of the absolute

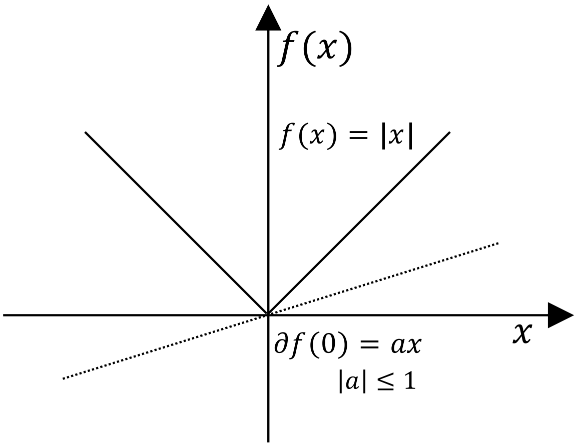


Figure 4.1: The subdifferential of absolute function at zero with dashed line as the subgradient.

function, $f(x) = |x|$, at zero is shown in Figure 4.1; any line that passes the origin with a slope between -1 and 1 is a subgradient of the absolute function. It is worth noting that the subdifferential operator is a point-to-set mapping; the equality relation is replaced by *"is a member of"* operator ($\in$). By considering the system (4.12) and using notion of subgradient, the subgradient flow system is then

$$\dot{x}(t) \in -\nabla f(x(t)) - \partial g(x(t)), \tag{4.15}$$

which can be discretized using the forward Euler discretization on $\nabla f(x(t))$ and backward Euler discretization on $\partial g(x(t))$ to obtain,

$$\frac{x_{k+1} - x_k}{\lambda} \in -\nabla f(x_k) - \partial g(x_{k+1}).$$

The terms related to x_k can be rearranged on the same side of the discretization as in (4.13) and the terms related to x_{k+1} can also be arranged on the other side of the discretization as in (4.14). We then achieve

$$x_{k+1} + \lambda \partial g(x_{k+1}) \ni x_k - \lambda \nabla f(x_k). \tag{4.16}$$

With notion of proximal operator, the relation (4.16) can be expressed as

$$x_{k+1} = \mathbf{prox}_{\lambda g}(x_k - \lambda \nabla f(x_k)), \tag{4.17}$$

which forms a fixed-point iteration. This is also known as **proximal gradient algorithm** or the forward-backward algorithm. However, it is often to encounter problems in the form of

$$\underset{x}{\text{minimize}} \;\; f(x) + g(Lx), \tag{4.18}$$

where proximal operator of $g(x)$ has a closed-form expression but may not for $\tilde{g}$ with $\tilde{g}(x) = g(Lx)$ for a general L. Solving (4.11) numerically in each iteration of proximal gradient is not feasible for a large scale setting. This is when ADMM algorithm (Alternating Direction Method of Multipliers, Boyd et al. (2011)) came into action. ADMM solves the problem in the form of

$$\begin{array}{ll} \text{minimize} & f(x) + \tilde{g}(z) \\ \text{subject to} & Ax + Bz = c. \end{array} \tag{4.19}$$

The example problem (4.18) can be solved by ADMM with $A = L, B = -I, c = 0$. The expression g is changed to $\tilde{g}$ to emphasize the difference between $g(x)$ and $\tilde{g}(z)$ since their argument is in different space for a non-square L.

One of the ADMM interpretations is the alternating minimization of augmented Lagrangian,

$$L_\rho(x, z, y) = f(x) + \tilde{g}(z) + y^T(c - Ax - Bz) + (\rho/2)\|c - Ax - Bz\|_2^2, \tag{4.20}$$

where x, z are both primal variables; z is referred to as an splitting variable; y is the dual variable. The algorithm parameter $\rho > 0$ plays an important role in the convergence of the algorithm which will be discussed later in this chapter. To keep the algorithm description simple, we changed the iteration index x_{k+1}, x_k, x_{k-1} to x^+, x, x^- respectively. ADMM simply updates the primal variable x and z that minimize the augmented Lagrangian in the alternating scheme and then updates the dual variable y as

$$x^+ = \underset{x}{\text{argmin}} \; L_\rho(x, z, y), \tag{4.21}$$

$$z^+ = \underset{z}{\text{argmin}} \; L_\rho(x^+, z, y), \tag{4.22}$$

$$y^+ = y + \rho(c - Ax^+ - Bz^+), \tag{4.23}$$

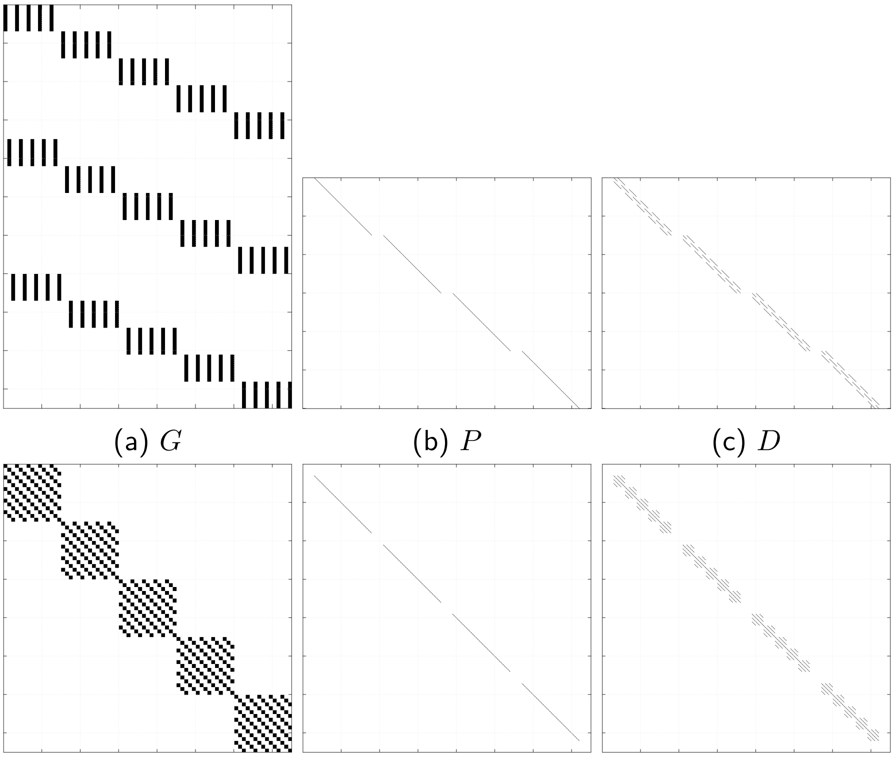
respectively. ADMM is regarded as the proximal algorithms because when $A = -B = I, c = 0$, the solution to problem (4.21), (4.22) are

$$\begin{array}{ll} \underset{x}{\text{argmin}} & f(x) + (\rho/2)\|x - (z - y/\rho)\|_2^2, \\ \underset{z}{\text{argmin}} & \tilde{g}(z) + (\rho/2)\|z - (x^+ - y/\rho)\|_2^2. \end{array}$$

These expressions are the evaluation of proximal operator, $\mathbf{prox}_{f/\rho}(z - y/\rho)$, $\mathbf{prox}_{\tilde{g}/\rho}(x^+ - y/\rho)$ in disguise.

4.2 Algorithms for convex formulations

The ADMM algorithm can be applied to solve (4.10) with convergence guaranteed under two conditions: the functions f, g are closed and proper convex functions and the unaugmented Lagrangian (without the quadratic term) (4.20) has a saddle point (Boyd et al., 2011). It is evident that cvx-CGN is already in the ADMM format (4.19) with $A = P, B = -I, c = 0$, but not for the DGN and FGN. To convert DGN, FGN into ADMM format, we split x and z in (4.19) such that $z_1 = L_1 x \in \mathbf{R}^{\tilde{m}_1}$ and $z_2 = L_2 x \in \mathbf{R}^{\tilde{m}_2}$. The function $f(x) = (1/2)\|Gx - b\|_2^2$



(a) G (b) P (c) D

(d) G^TG (e) P^TP (f) D^TD

Figure 4.2: Sparsity pattern of related matrices when $n = 5, p = 10, K = 3$.

is convex and its gradient is Lipschitz continuous. With the choice of splitting, we re-define the penalty g from (4.7)-(4.9) to $\tilde{g} : \mathbf{R}^{\tilde{m}_1} \times \mathbf{R}^{\tilde{m}_2} \to \mathbf{R}$, $\tilde{g}(z_1, z_2) = \lambda_1 h(z_1; \mathcal{B}_1) + \lambda_2 h(z_2; \mathcal{B}_2)$. Our ADMM format (4.19) corresponds to

$$B = -I,\ A = \begin{bmatrix} L_1 \\ L_2 \end{bmatrix},\ \text{where} \begin{bmatrix} L_1 \\ L_2 \end{bmatrix} = \begin{bmatrix} P \\ P \end{bmatrix} \text{ for DGN, and } \begin{bmatrix} L_1 \\ L_2 \end{bmatrix} = \begin{bmatrix} P \\ D \end{bmatrix} \text{ for FGN.} \quad (4.24)$$

As stated before, the choice of ρ strongly affects the convergence speed of the algorithm. The following content is a variant of ADMM that update the penalty parameter ρ as the algorithm goes. We also discuss the other choices of the algorithms later in this section.

Spectral ADMM

We consider ADMM algorithm with spectral penalty parameter update rule proposed in Xu et al. (2017b). According to our problem parameters, the augmented Lagrangian is $L_\rho(x, z, y) = f(x) + \tilde{g}(z_1, z_2) + y^T(z - Ax) + (\rho/2)\|z - Ax\|_2^2$. By following the updates in (4.21)-(4.23), the x-update step involves the minimization of L_ρ over x which satisfies the zero-gradient condition:

$$(\rho A^T A + G^T G)x = G^T b + A^T y + \rho A^T z.$$

The computation complexity per iteration is primarily come from solving the linear equation. The worst-case computation complexity is $\mathcal{O}((n^2pK)^3)$. By seeing that $\rho A^TA + G^TG$ has a block diagonal structure as shown in Figure 4.2, we can further exploit by solving each block separately. The structure of the system of equations are separated into n smaller system of equations, each with size of $npK \times npK$. Therefore, we can reduce the computation complexity from $\mathcal{O}((n^2pK)^3)$ to $\mathcal{O}(n(npK)^3)$. Moreover, we can use the Cholesky factorization to further reduce the computation complexity. We emphasize that the factorization is only computed when the penalty ρ is updated. When ρ does not change, the computation complexity per iteration is reduced to $\mathcal{O}(n(npK)^2)$.

As the definition of h is associated with a partition being used, we denote $h_1(x) := h(x; \mathcal{B}_1)$ and $h_2(x) := h(x; \mathcal{B}_2)$ for the simplicity of notation. The z-update step is to minimize L_ρ over z and takes the form of the proximal operator (Parikh and Boyd, 2014):

$$\begin{aligned}\begin{bmatrix} z_1^+ \\ z_2^+ \end{bmatrix} = \operatorname*{argmin}_z \ \tilde{g}(z_1, z_2) + (\rho/2) \left\| \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} - \begin{bmatrix} L_1x - y_1/\rho \\ L_2x - y_2/\rho \end{bmatrix} \right\|_2^2 &= \mathbf{prox}_{\tilde{g}/\rho} \left(\begin{bmatrix} L_1x - y_1/\rho \\ L_2x - y_2/\rho \end{bmatrix} \right) \\ &= \begin{bmatrix} \mathbf{prox}_{\lambda_1 h_1/\rho}(L_1x - y_1/\rho) \\ \mathbf{prox}_{\lambda_2 h_2/\rho}(L_2x - y_2/\rho) \end{bmatrix}.\end{aligned}$$

The last equality follows directly from the separable summation property of $\tilde{g}$ (Parikh and Boyd, 2014, §2). The functions h_1 and h_2 all take the form of a composite of weighted-ℓ_q and ℓ_2 norms, expressed as $h(x; \mathcal{B}) = \sum_{l \in \mathcal{B}} w_l \|x_l\|_2^q$ given in (4.5). It is well-known that for $q = 1$, the proximal operator of $h(x; \mathcal{B})$ is the *weighted block-soft thresholding* (Parikh and Boyd, 2014, §6.5). For all $l \in \mathcal{B}$,

$$(\mathbf{prox}_{\alpha h}(u))_l = (1 - \alpha w_l/\|u_l\|_2)_+ \cdot u_l = \begin{cases} (1 - \alpha w_l/\|u_l\|_2)u_l, & \|u_l\|_2 \geq \alpha w_l, \\ 0, & \|u_l\|_2 < \alpha w_l. \end{cases} \tag{4.25}$$

The proximal operators (4.25) can be computed in parallel for each partition in the partition set given in (4.3), (4.4).

The ADMM algorithm for solving (4.10) is named **SparseGrangerNet** and now presented in Algorithm 1. From (Boyd et al., 2011, §3.3), it is worth noting that the critical point of (4.19) must satisfy

$$0 = -Ax + z, \tag{4.26}$$

$$0 = \nabla f(x) - A^Ty, \tag{4.27}$$

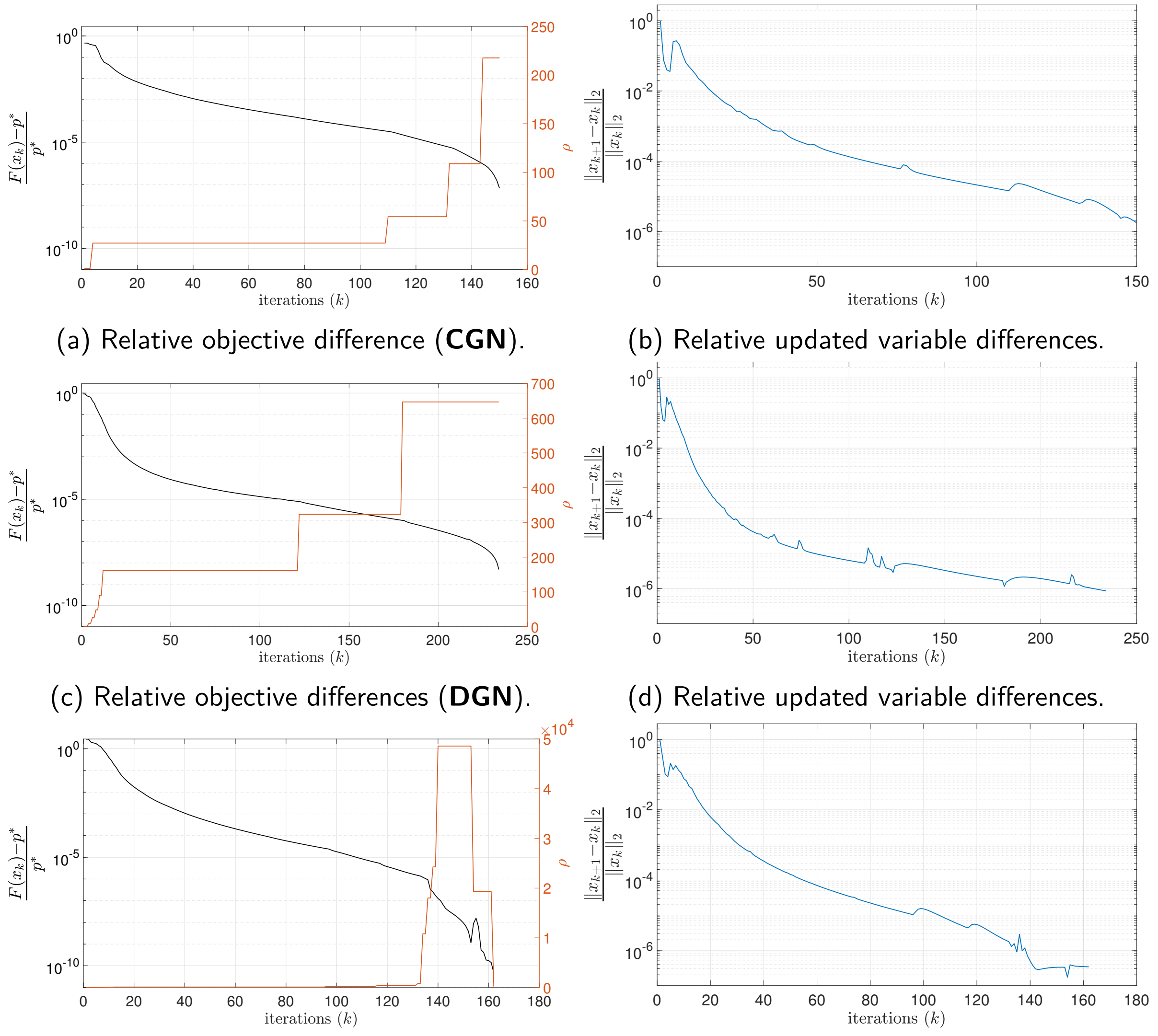
$$0 \in \partial \tilde{g}(z_1, z_2) + y, \tag{4.28}$$

where (4.26) is called primal feasibility condition since it is the constraint of ADMM in (4.19); (4.27), (4.28) are called the dual feasiblity since they are zero (sub)gradient condition of the unaugmented Lagrangian. The iterates generated from ADMM do not satisfy these condition since x-update step (4.21) and z-update step (4.22) are the solutions to

$$\begin{aligned} 0 &= \nabla f(x^+) - A^Ty^+ + \rho A^T(z^+ - z), \\ 0 &\in \partial \tilde{g}(z_1^+, z_2^+) + y^+, \end{aligned}$$

respectively. It is obvious that the z-update step makes the iterate satisfies optimality condition in (4.28) but not for x-update step since it has an extra term, $s = \rho A^T(z^+ - z)$. The extra term is called the dual residual. Other than the dual residuals, the term $r = -Ax^+ + z^+$ is called the primal residual and it measures the primal feasibility of the problem. After updating x, z and y (dual variable), the primal and dual residuals (r, s) are computed. We follow the stopping criterion on these two residuals given in (Boyd et al., 2011, §3.3.1) where the absolute tolerance (ϵ_{abs}), relative tolerance (ϵ_{rel}) are set to 10^{-7} and 10^{-5}, respectively. We also implement ρ-update rules for every T iteration because it is known that the penalty parameter (ρ) greatly affects the algorithm convergence.

The adaptive rule presented in Subroutine 2 follows the spectral penalty selection proposed by Xu et al. (2017b). The rule was inspired by the Barzilai-Borwein (BB) gradient method that approximated the secant condition in smooth unconstrained problems. The adaptive rule was brought into ADMM in Xu et al. (2017b) with a safeguard step for measuring the goodness of fit for linear approximations of subgradients of dual ADMM objective that was split into two terms according to the conjugate of f and g. The linear approximation of each term was parameterized by two choices of spectral step sizes: steepest descent and minimum gradient, and some hybrid rules were further applied to determine the step size. When the linear approximations were sufficiently credible (as measured by correlations), the penalty was updated as the geometric mean or one of the step sizes; otherwise, the previous ρ was kept for the next iteration. We illustrated the performance of the spectral ADMM when solving CGN, DGN and FGN in Figure 4.3. Since convergence rate of ADMM is $\mathcal{O}(1/k)$, we are expected to see a negative log trend in a log-scale plot of the relative difference between the objective value and the optimal objective value. From the figure, it can be seen that the relative difference, when the penalty is unchanged, follow the negative log trend as expected in the fixed penalty ADMM.



(a) Relative objective difference (**CGN**).

(b) Relative updated variable differences.

(c) Relative objective differences (**DGN**).

(d) Relative updated variable differences.

(e) Relative objective differences (**FGN**).

(f) Relative updated variable differences.

Figure 4.3: Spectral ADMM performance on CGN, DGN and FGN: $F(x)$ denotes the objective function in (4.10) with $q = 1$ (convex case), and p^* denotes the optimal value. The problem and estimation parameters are $n = 20, p = 2, K = 5, T = 100$.

Other available algorithms

There are other alternative approaches to solve our convex formulations as in the following.

Proximal gradient methods. There are other algorithms that can solve cvx-CGN formulation. Even though the regularization term of CGN is composited with P but the proximal operator of $\tilde{g}(x) = g(Px)$ has a closed-form expression because P is a projection matrix. Therefore, the proximal gradient methods and the accelerated proximal gradient (Parikh and Boyd, 2014, §4.3) can also be used for CGN.

Algorithm 1: SparseGrangerNet

```
Problem parameters: $A = \begin{bmatrix} L_1 \\ L_2 \end{bmatrix}, G, b$
Algorithm parameters: $T, \epsilon_{\text{pri}}, \epsilon_{\text{dual}}$
Updating sequences: $x, y = (y_1, y_2), z = (z_1, z_2), (x, y, z)_{\text{cached}}, \rho > 0, k = 1$
while $\|r\|_2 \geq \epsilon_{\text{pri}}$ and $\|s\|_2 \geq \epsilon_{\text{dual}}$ do          // stopping criterion
    $x^+ = (\rho A^T A + G^T G)^{-1} \left( G^T b + A^T (y + \rho z) \right)$
    $z_1^+ = \mathbf{prox}_{\lambda_1 h_1/\rho}(L_1 x^+ - y_1/\rho)$      // thresholding with partition $\mathcal{B}_1$
    $z_2^+ = \mathbf{prox}_{\lambda_2 h_2/\rho}(L_2 x^+ - y_2/\rho)$      // thresholding with partition $\mathcal{B}_2$
    $y^+ = y + \rho(z^+ - Ax^+)$
    $r = z^+ - Ax^+$                                                       // primal residual
    $s = \rho A^T (z^+ - z)$                                               // dual residual
    if mod $(k, T) = 0$ then                                   // Update $\rho$ every $T$ iterations
        $\rho^+ = \text{UpdatePenalty}(\cdot)$
    else
        $\rho^+ = \rho$
    $k \leftarrow k + 1$
```

SDMM. We emphasize that the technique of splitting $A = (L_1, L_2)$ is not new. The SDMM algorithm (Simultaneous direction method of multipliers) proposed by Combettes and Pesquet (2011) is similar to Algorithm 1 but has been derived from a different point of view on the objective function. The SDMM solve the problem in the form of

$$\begin{array}{ll} \text{minimize} & g_0(z_0) + g_1(z_1) + g_2(z_2) \\ \text{subject to} & Ax + Bz = c, \end{array} \tag{4.29}$$

where $z_0 = L_0 x \in \mathbf{R}^{nNK}, z_1 = L_1 x \in \mathbf{R}^{\tilde{m}_1}, z_2 = L_2 x \in \mathbf{R}^{\tilde{m}_2}$. The objective function of (4.29) is separable in z_0, z_1, z_2. Our formulations, (4.10), can be arranged to the form (4.29) by setting,

$$B = -I, \;\; A = \begin{bmatrix} L_0 \\ L_1 \\ L_2 \end{bmatrix}, \text{ where } \begin{bmatrix} L_0 \\ L_1 \\ L_2 \end{bmatrix} = \begin{bmatrix} G \\ P \\ P \end{bmatrix} \text{ for DGN, and } \begin{bmatrix} L_0 \\ L_1 \\ L_2 \end{bmatrix} = \begin{bmatrix} G \\ P \\ D \end{bmatrix} \text{ for FGN,} \tag{4.30}$$

with $g_0(z_0) = (1/2)\|z_0 - b\|_2^2$ and g_1, g_2 are the regularization terms in DGN or FGN. Therefore,

Subroutine 2: UpdatePenalty($\cdot$) for **convex** formulations: Spectral adaptive ρ (Xu et al., 2017b)

```
Problem parameters: $A = \begin{bmatrix} L_1 \\ L_2 \end{bmatrix}, x^+, (x, y, z), (x, y, z)_{\text{cached}}$
Algorithm parameters: $\epsilon_c$
Updating sequences: $\rho > 0$
$\hat{y} = y + \rho(z - Ax^+), \Delta\hat{y} = \hat{y} - \hat{y}_{\text{cached}}$
$\Delta F = A(x - x_{\text{cached}})$          // $\Delta$ subdifferential of dual obj. from $f$
$a_1 = \frac{\Delta F^T \Delta\hat{y}}{\|\Delta F\|_2^2}, a_2 = \frac{\|\Delta\hat{y}\|_2^2}{\Delta F^T \Delta\hat{y}}$          // a1: minimum gradient, a2:steepest
 descent
if $2a_1 > a_2$ then // choose spectral step size for $\Delta F$
    $a = a_1$
else
    $a = a_2 - 0.5a_1$
$\Delta y = y - y_{\text{cached}}$,
$\Delta G = -(z - z_{\text{cached}})$          // $\Delta$ subdifferential of dual obj. from $g$
$b_1 = \frac{\Delta G^T \Delta y}{\|\Delta G\|_2^2}, b_2 = \frac{\|\Delta y\|_2^2}{\Delta G^T \Delta y}$ // b1: minimum gradient, b2:steepest descent
if $2b_1 > b_2$ then // choose spectral step size for $\Delta G$
    $b = b_1$
else
    $b = b_2 - 0.5b_1$
$c_1 = \frac{\Delta F^T \Delta\hat{y}}{\|\Delta F\|_2 \|\Delta\hat{y}\|_2}, c_2 = \frac{\Delta G^T \Delta y}{\|\Delta G\|_2 \|\Delta y\|_2}$          // correlation terms: linear
 approximations of $\Delta F, \Delta G$
 /* Safeguard update rule for $\rho$                              */
if $c_1 > \epsilon_c$ and $c_2 > \epsilon_c$ then // use geometric mean when high
 correlations
    $\rho^+ = \sqrt{ab}$
else if $c_1 > \epsilon_c$ and $c_2 \leq \epsilon_c$ then
    $\rho^+ = a$
else if $c_1 \leq \epsilon_c$ and $c_2 > \epsilon_c$ then
    $\rho^+ = b$
else
    $\rho^+ = \rho$
```

the SDMM iterations are

$$
\begin{aligned}
x^+ &= (\rho A^T A)^{-1} A^T (y + \rho z) = (A^T A)^{-1} A^T (z + y/\rho), \\
z_0^+ &= \mathbf{prox}_{g_0}(Gx - y_0/\rho), \\
z_1^+ &= \mathbf{prox}_{g_1}(L_1 x - y_1/\rho), \\
z_2^+ &= \mathbf{prox}_{g_2}(L_2 x - y_2/\rho), \\
y^+ &= y + \rho(z^+ - Ax^+).
\end{aligned}
$$

By comparing these iterations to Algorithm 1, there is no difference in computation cost since the proximal operators have closed-form expression. However, when exploiting the x-update step by using Cholesky factorization, SDMM does not require to repeat the factorization after updating penalty parameter as in Algorithm 1. This is because ρ can be factored out in SDMM but not for Algorithm 1. It is worth noting that the length of z_0 is proportional to the length of time-series (*i.e.* matrix G has nNK rows with N as the effective time-points), causing additional memory usage when compared to Algorithm 1.

4.3 Algorithms for non-convex formulations

Since the convergence of either ADMM or spectral ADMM on our non-convex problems is still unknown, we propose a heuristic ρ update rule that makes ADMM converged for our non-convex problems in practice. After the algorithm is introduced, we reviewed related issues on the convergence analysis of the ADMM applied to non-convex problems with certain structures in literature. An important tool in convergence analysis of non-smooth non-convex optimization problems that satisfied the Kurdyka - Łojasiewicz (KL) inequality which the objective is called the KL function; see Bolte et al. (2009) and the references therein. The convergence analysis is useful for modifying the existing algorithms to ensure a convergence for optimizing KL functions. We emphasize that all of our problem formulations are KL functions according to the appendix of Feng et al. (2020). The proximal gradient algorithm in (4.17) is globally converged (Attouch et al., 2013) if g is a KL function. For CGN, the proximal gradient algorithm can be used but may suffer from a slow convergence. The accelerated proximal gradient algorithm does not have a convergence guarantee for non-convex CGN. The issue of accelerated proximal gradient is solved by Li and Lin (2015). They proposed the non-monotone accelerated proximal gradient algorithm (nmAPG) which is adapted from accelerated proximal gradient with a safeguard rule to ensure a global convergence. This topic will be discussed later in this section.

The proximal algorithms in this section are primarily based on the evaluation of the proximal operator of the weighted $\ell_{2,1/2}$ penalty (Hu et al., 2017). By following the notation from (4.25), the proximal operator of $h(x;\mathcal{B}) = \sum_{l\in\mathcal{B}} w_l \|x_l\|_2^{1/2}$ takes the form of

$$
(\mathbf{prox}_{\alpha h}(u))_l = \begin{cases} \left(\frac{16\|u_l\|_2^{3/2} \cos^3(R(u_l))}{3\sqrt{3}\alpha w_l + 16\|u_l\|_2^{3/2} \cos^3(R(u_l))} \right) u_l & \|u_l\|_2 > \frac{3}{2}(\alpha w_l)^{2/3}, \\ 0, & \|u_l\|_2 \leq \frac{3}{2}(\alpha w_l)^{2/3}, \end{cases} \tag{4.31}
$$

where $R(u_l) = \pi/3 - (1/3)\,\text{arccos}(\frac{\alpha w_l}{4}(3/\|u_l\|_2)^{3/2})$, for all $l \in \mathcal{B}$.

Although the proximal gradient method and nmAPG have a global convergence for both DGN and FGN, it is infeasible in a large-scale setting since the proximal operator of $g(x)$ in (4.8) and (4.9) does not have a closed-form expression. For this reason, Algorithm 1 is more favorable to solve DGN and FGN in a large scale setting. We will discuss on the computational cost comparison later in this section.

In the following, we first discuss our proposed adaptive ADMM scheme that converges in practice. We also provide a scheme to numerically compute the proximal operator in the case that the proximal gradient algorithm is used in DGN. Then we discuss the technique used in the nmAPG algorithm to make the acceleration on non-convex problem possible. In the end, we also discuss the other choices of proximal algorithms.

Adaptive ADMM

There is no guarantee for a convergence of ADMM with a fixed ρ when solving non-convex problems in general. However, we observed that when ρ is too large, the primal residual has a fast convergence but the dual residual is slowly converged; if ρ is too small, the iterations could diverge. Therefore, we come up with a strategy that the penalty ρ is increased and stops adapting after the primal residuals converge to avoid a slow convergence from ρ being too large. This heuristic update step is described in Subroutine 3; we start ρ with a small value and increase it by a factor of 2 every T iteration. After the primal residual converges, we stop the penalty update scheme. This rule was also proposed for solving convex problems in Xu et al. (2017a) as LA-ADMM, with an improved iteration complexity from a fixed-penalty scheme, where the choice of initial ρ depends on properties of the objective function. Unlike Subroutine 3, the scheme of Xu et al. (2017a) has no termination rule; ρ can increase to a large value, leading to a slow convergence.

Subroutine 3: UpdatePenalty(·) for **non-convex** formulations

```
Problem parameter : $r$
Algorithm parameter: $\epsilon_{\text{pri}}$
Updating sequences: $\rho > 0$
if $\|r\|_2 \geq \epsilon_{\text{pri}}$ then
    $\rho^+ = 2\rho$
else
    $\rho^+ = \rho$
```

Convergence of ADMM for non-convex problems. ADMM convergence analysis in non-convex problems mostly depends on properties of matrix A, B in (4.24). The analysis from Li and Pong (2015); Wang et al. (2018); Zhang et al. (2016) assumed A to be a full row rank matrix in their convergence analysis. This condition is impossible for DGN and FGN to satisfy since the matrix A in (4.24) is a tall matrix. A weaker condition of matrix A for

ADMM convergence analysis is found in Wang et al. (2019). They required only **range**$(B) \subset$ **range**(A), instead of full row rank A. However, both DGN and FGN still cannot satisfy their assumptions. The row rank assumption can be explained from subdifferential calculus (Rockafellar and Wets, 1998). The subdifferential chain rule required a full row rank A to have the equality[1]

$$\partial(g \circ A)(x) = A^T \partial g(Ax).$$

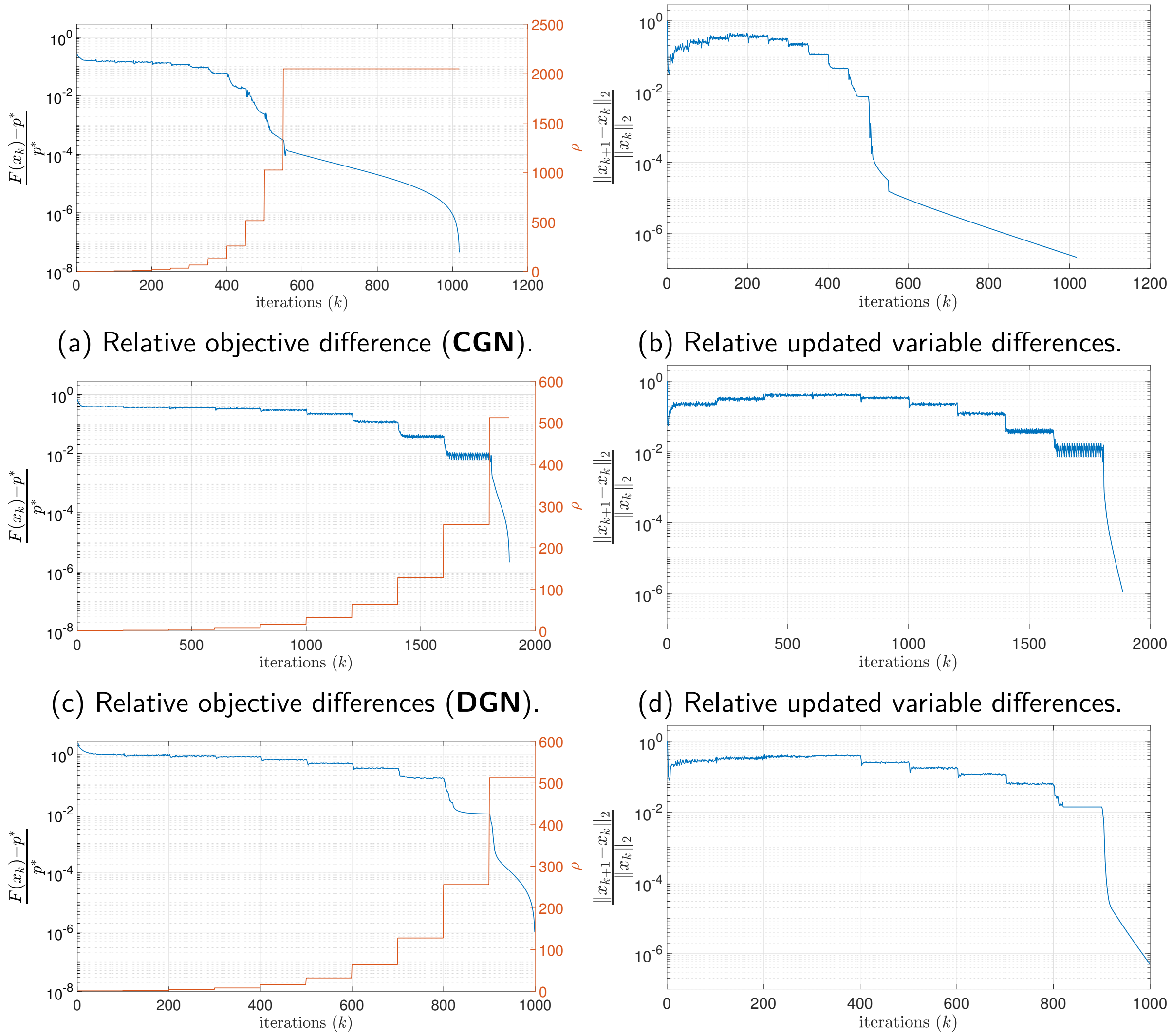
This is the main obstacle for the convergence analysis of ADMM for non-smooth and non-convex problems. When we explore broader types of Lagrangian-based algorithms, one of which is ADMM, a unified treatment of convergence analysis was reviewed in Sabach and Teboulle (2019). A recent *adaptive Lagrangian-based multiplier (ALBUM)* method (Sabach and Teboulle, 2019) for non-convex composite problems relies on the so-called *a uniform regularity* condition of the composite mapping, which essentially says, in our case, that A must be surjective, similar to the full rank assumption of A in Li and Pong (2015). To the best of our knowledge, the undesirable property of our A has become the main obstacle to analyze the convergence of ADMM when applied to DGN and FGN. We leave this as an open problem, while our implementation (with fine-tuned parameters) of adaptive ADMM (Algorithm 1 with Subroutine 3) to DGN and FGN did not return divergent instances in our experiments.

The performance example of the adaptive ADMM when solving all non-convex formulations is presented in Figure 4.4. For CGN, it can be seen that after around 500 iterations, the oscillation stopped and converged. When the penalty ρ is updated, the objective is abruptly changed. We observed that, when the penalty parameter is too small, the sequences oscillated without pattern; when ρ is sufficiently large, the sequences oscillated with a repeating pattern, and then the algorithm converges if the penalty is further increased thereafter. This may be used as exploitation for convergence detection.

Non-monotone accelerated proximal gradient algorithm (nmAPG) for CGN

We applied an accelerated variant of the proximal gradient algorithm called nmAPG proposed by Li and Lin (2015) presented in Algorithm 4 to solve non-convex CGN with convergence guaranteed. The convergence of the algorithm is based on the controlling of the energy function to be a monotonic decrease sequence. The energy function is simply a weighted average of objective value from the current iteration and to the past iterations with exponentially decaying weights. The decaying rate is controlled by parameter $\eta \in [0, 1)$. The algorithm simply selects the step that gives a sufficiently lower energy function from the proximal gradient step and accelerated proximal gradient. The monitoring scheme speeds up the convergence by accelerated proximal gradient algorithm while ensuring the global convergence by using the proximal gradient as a safeguard step. They also provided a Barzilai-Borwein (BB) backtracking line search to achieve a larger proximal step size. The idea behind the BB line search is to estimate the curvature of the update step by choosing step size α in the proximal step to approximate a secant condition in a least-square sense. If the sufficient descent of a proximal

[1] See Theorem 10.6 in Rockafellar and Wets (1998), Corollary 2.52 in Mordukhovich and Nam (2013)



(a) Relative objective difference (**CGN**).

(b) Relative updated variable differences.

(c) Relative objective differences (**DGN**).

(d) Relative updated variable differences.

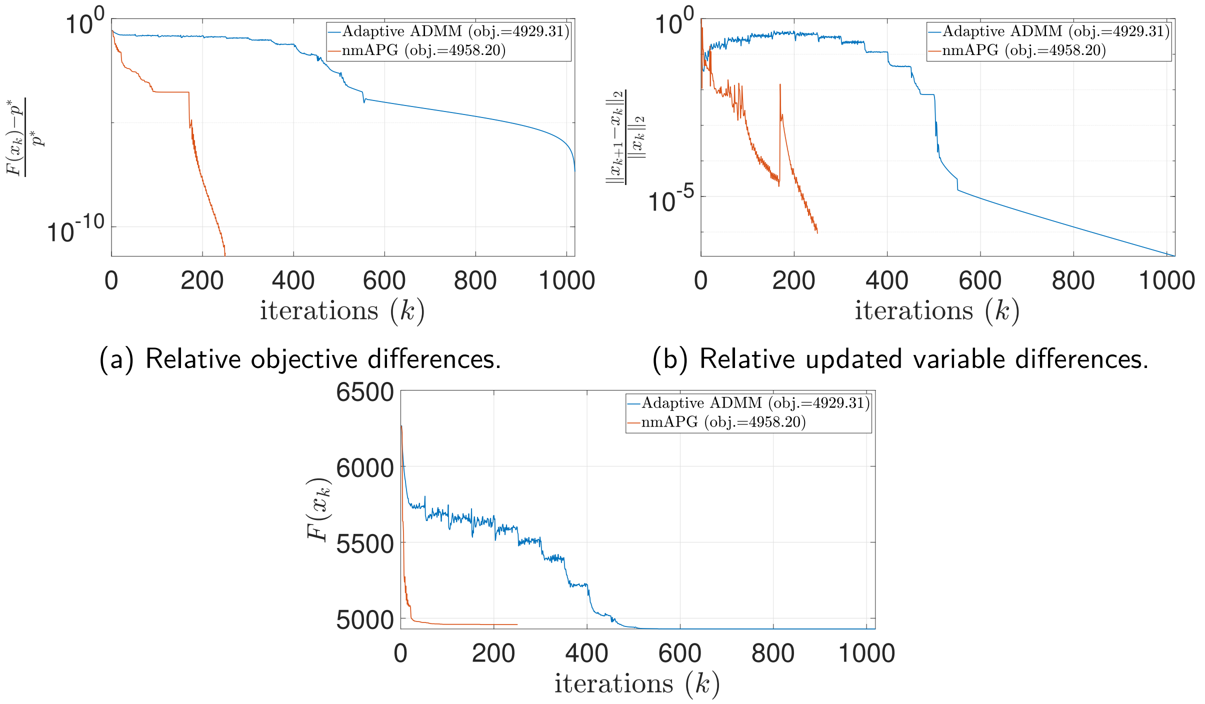
(e) Relative objective differences (**FGN**).

(f) Relative updated variable differences.

Figure 4.4: Adaptive ADMM performance on CGN, DGN and FGN: $F(x)$ denotes the objective function in (4.10) with $q = 1/2$ (non-convex case), and p^* denotes the optimal value. The problem and estimation parameters are $n = 20, p = 2, K = 5, T = 100$.

step is not satisfied, the step size is scaled down with a factor of ρ. The line search rule is described in Subroutine 5, 6. It is also worth noting that the *non-monotone* in the name comes from the non-monotone in objective function $F(x) = (1/2)\|Gx - b\|_2^2 + g(x)$ with $g(x)$ in (4.7) but the energy function is actually a monotonic decreasing sequences.

We also compare the performance of ADMM and nmAPG for solving non-convex CGN in Figure 4.5. The nmAPG algorithm converged at a lower iteration than adaptive ADMM with the steeper relative objective curve. This indicated a higher convergence rate of nmAPG over adaptive ADMM. Slow convergence of adaptive ADMM may come from the initial penalty ρ is too small. Since the problem to solve is non-convex, it is interesting why the adaptive ADMM



(a) Relative objective differences.

(b) Relative updated variable differences.

(c) Objective differences.

Figure 4.5: Performance comparison between nmAPG and adaptive ADMM when solving CGN formulation with $(n = 20, p = 2, K = 5, T = 100)$

converged to a lower objective value when compared to nmAPG in almost all instances. From this favorable property in practice, we primarily use adaptive ADMM in the experiment.

Proximal gradient algorithm for CGN

As stated before, the proximal gradient algorithm in (4.17) can be applied on CGN with global convergence. The update scheme is simply plugging the loss gradient $\nabla f(x) = G^T(Gx - b)$ into (4.17) to obtain,

$$x^+ = \mathbf{prox}_{\alpha g}(x - \alpha G^T(Gx - b)),$$

with $\alpha < 1/\|G\|_2$; $g(x)$ in (4.7). The global convergence of the proximal gradient method when solving our problem in the same class with ours is provided in Attouch et al. (2013). However, the algorithm has a slower convergence when compared to nmAPG.

Inexact proximal algorithms for DGN formulation

Even though the proximal gradient algorithm in (4.17) and nmAPG are globally converged for all of our formulations, it is inefficient to apply them to solve DGN formulations due to the lack of closed-form expression for the proximal operator. To use nmAPG to solve DGN, we are required to solve (4.11) numerically. However, the numerical solution to (4.11) may not produce a sparse solution. Therefore, a thresholding scheme must be introduced.

Algorithm 4: nmAPG (Li and Lin, 2015)

```
Problem parameters: $G, b$
Algorithm parameters: $\alpha_x \leq 1/\|G\|_2, \alpha_y \leq 1/\|G\|_2, \eta \in [0,1), \delta > 0$
Updating sequences: $z = x = x^- = x_0, t = 1, t^- = 0, q = 1$
$F(x) = (1/2)\|Gx - b\|_2^2 + g(x)$ with $g(x)$ in (4.7)
while $\|x^+ - x\|_2 \geq \epsilon \|x\|_2$ do
    $y = x + \frac{t^-}{t}(z - x) + \frac{t^- - 1}{t}(x - x^-)$, // Caching momentum
    $z^+ = \textbf{prox}_{\alpha_y g}(y - \alpha_y G^T(Gy - b))$, // Replace with Subroutine 5
    if $F(z^+) \leq c - \delta\|z^+ - y\|_2^2$ then // Check descent of accelerated step
        $x^+ = z^+$
    else
        $v^+ = \textbf{prox}_{\alpha_x g}(x - \alpha_x G^T(Gx - b))$, // Replace with Subroutine 6
        /* Select proximal gradient step if acceleration fail */
        $x^+ = \begin{cases} z^+, \textbf{if } F(z^+) \leq F(v^+), \\ v^+, \textbf{else}, \end{cases}$ ,
        $t^+ = (1/2)(1 + \sqrt{4t^2 + 1})$,
        /* Recursively compute energy function $c^+$ */
        $q^+ = \eta q + 1$, $c^+ = (\eta q c + F(x^+))/q^+$.
```

Subroutine 5: BB line search for α_y

```
Given $s_y = y - y^-, r = G^T G s_y, \alpha_y = \frac{\|s_y\|_2^2}{s_y^T r}$, $0 < \rho < 1, \delta > 0$
$F(x) = (1/2)\|Gx - b\|_2^2 + g(x)$ with $g(x)$ in (4.7)
while $F(z^+) \geq F(y) - \delta\|z^+ - y\|_2^2$, and $F(z^+) \geq c - \delta\|z^+ - y\|_2^2$, do
    $z^+ = \textbf{prox}_{\alpha_y g}(y - \alpha_y G^T(Gy - b))$,
    $\alpha_y = \rho\alpha_y$,
```

Subroutine 6: BB line search for α_x

```
Given $s_x = x - y^-, r = G^T G s_x, \alpha_x = \frac{\|s_x\|_2^2}{s_x^T r}$, $0 < \rho < 1, \delta > 0$
$F(x) = (1/2)\|Gx - b\|_2^2 + g(x)$ with $g(x)$ in (4.7)
while $F(v^+) \geq c - \delta\|v^+ - x\|_2^2$ do
    $v^+ = \textbf{prox}_{\alpha_x g}(x - \alpha_x G^T(Gx - b))$,
    $\alpha_x = \rho\alpha_x$,
```

By recognizing that the problem (4.11), when using $g(x)$ in (4.8), is separable with partition set $\mathcal{K}$ in (4.4). We compute the proximal operator by solving (4.11) in parallel by each the partition of size pK as,

$$\underset{x}{\text{minimize}} \;\; \lambda_1 \sum_{l=1}^{K} w_l \|x_l\|_2^{1/2} + \lambda_2 v \|x\|_2^{1/2} + (1/2)\|x - z\|_2^2, \tag{4.32}$$

with $x_l \in \mathbf{R}^p, x = (x_1, \ldots, x_K) \in \mathbf{R}^{pK}$. By assuming all $x_l \neq 0$, we obtain the zero gradient conditions as

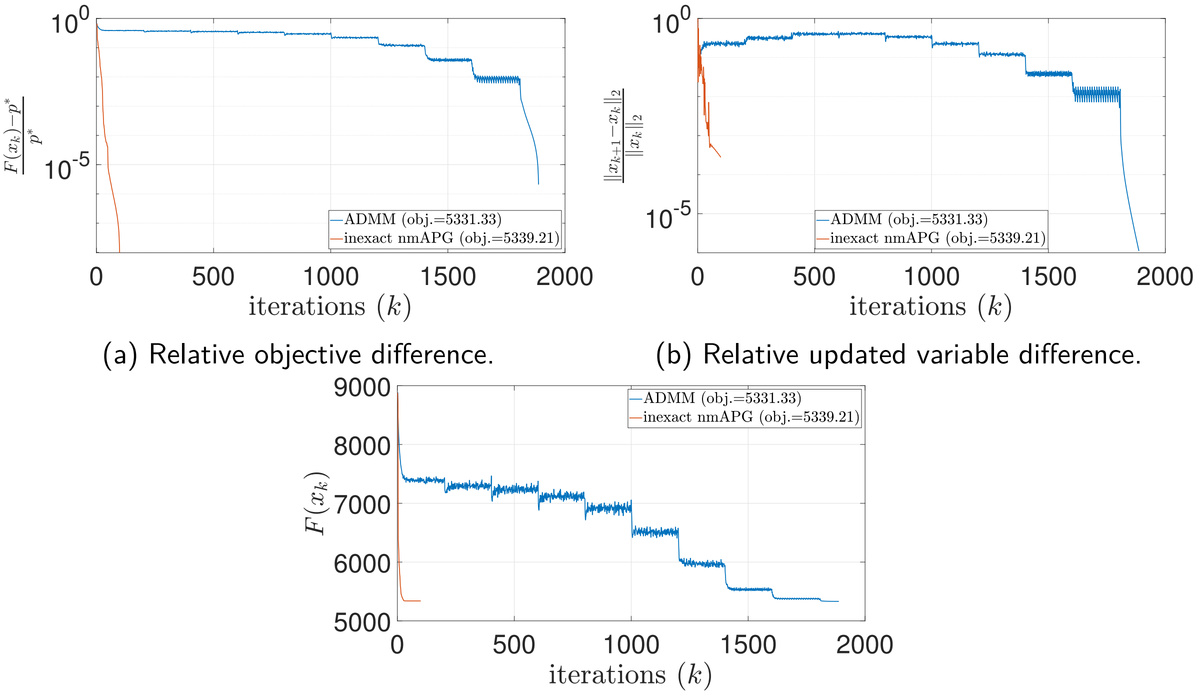
$$\left(1 + \frac{\lambda_1 w_l/2}{\|x_l\|_2^{3/2}} + \frac{\lambda_2 v/2}{\|x\|_2^{3/2}}\right) x_l = z_l, \quad l = 1, \ldots, K. \tag{4.33}$$

We heuristically solve the (4.33) using a fixed point iteration,

$$x_l^+ = z_l - \left(\frac{\lambda_1 w_l/2}{\|x_l\|_2^{3/2}} + \frac{\lambda_2 v/2}{\|x\|_2^{3/2}}\right) x_l, \quad l = 1, \ldots, K. \tag{4.34}$$

We still do not know whether the fixed point iteration (4.34) is converged but it converged in all of our numerical examples. Since the fixed-point iteration is derived under non-zero conditions, we must determine the correct sparsity pattern first, and update the fixed-point iteration only for the non-zero parts. Therefore, we used fixed-point iteration with every possible combination of the non-zero patterns. The pattern with lowest proximal objective (4.11) is selected to be a solution of (4.32). As a remark, the number of (4.34) computation is 2^K per proximal operator call and this is only a sub-block of size pK of the entire vector of size $(n^2 - n)pK$, which is to solve (4.32) for $(n^2 - n)2^K$ times. This method is feasible only for small K.

We illustrated the performance difference between adaptive ADMM (Algorithm 3) and the nmAPG with the inexact proximal operator (or inexact nmAPG) for solving DGN in Figure 4.6. It can be seen that the inexact nmAPG converged much faster than the proposed adaptive ADMM in the sense of using a lower number of iterations. However, the time usage in adaptive ADMM is significantly lower than that of the inexact nmAPG. This is because the inexact nmAPG has to solve (4.33) repeatedly leading to a high computational cost each time the proximal operator is called. Moreover, when applying the BB line-search subroutine to the inexact nmAPG, the proximal operator may be required to be evaluated multiple times for a single iteration, leading to an even higher computational cost. The computation cost can be reduced by lowering the tolerance of the fixed point iterations; however, numerical errors affect the convergence property of the algorithm. In Gu et al. (2018), Yao et al. (2017), they proposed a method to control the numerical error when solving (4.11) up to some tolerance degree to make the nmAPG algorithm converged. This can be an alternative algorithm other than the adaptive ADMM algorithm for solving non-convex DGN. However, we do not know the convergence of the fixed-point iteration (4.34), so its convergence property is still unknown as for adaptive ADMM. These reasons make the adaptive ADMM be a more suitable choice than nmAPG to solve DGN.



(a) Relative objective difference.

(b) Relative updated variable difference.

(c) Objective value difference.

Figure 4.6: Algorithm performance comparison between inexact nmAPG and adaptive ADMM when solving DGN. $F(x)$ denotes the objective function in (4.10) with $q = 1/2$ (non-convex case), and p^* denotes the optimal value. The problem and estimation parameters are $n = 20, p = 2, K = 5, T = 100$.

Other proximal methods

As a remark, there may be other classes of algorithms that used to solve the problems in a similar class with our formulations; however, our formulation structure violated their assumption. Themelis and Patrinos (2020) analyzed the convergence of a variant of ADMM called the relaxed ADMM algorithm on our problem class but still requires the full row rank assumption. Qiao et al. (2016) analyzed the convergence of a variant of ADMM called the linearized ADMM on our class of problem which still has the full row rank assumption to guarantee the global convergence. In most of the literature, the convergence of our problem is still an open question. We emphasize that there is no divergent instance in our results, which will be presented in Chapter 5. Therefore, the convergence of ADMM with our heuristic penalty update rule may be an indication that the rank assumption may be too restrictive.

Chapter V

EXPERIMENTAL RESULTS

We demonstrated the performance of our proposed methods in this chapter. We set up simulation experiments to show the advantages of the proposed methods over others under various circumstances. This chapter consists of three parts. The first part concerns on the generation of stable ground-truth VAR models with predefined prior of model relations described in Chapter 3. We empirically compared our methods with the existing works on extensive simulations. In the last part of this chapter, we demonstrated the application of our formulations as the classification task and brain connectivity differences learning between children with attention deficit hyperactivity disorder (ADHD) and the typically developing children (TDC). We provide the source codes to all of our experiments in https://github.com/parinthorn/JGranger_ncvx/.

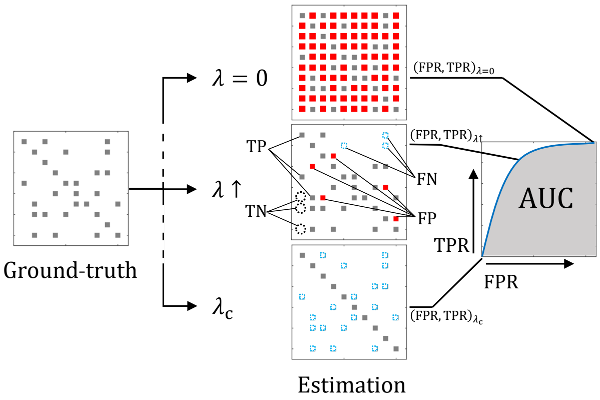

$\lambda = 0$
$\lambda \uparrow$
λ_c
Ground-truth
TP
TN
FN
FP
$(\mathrm{FPR}, \mathrm{TPR})_{\lambda=0}$
$(\mathrm{FPR}, \mathrm{TPR})_{\lambda\uparrow}$
$(\mathrm{FPR}, \mathrm{TPR})_{\lambda_c}$
AUC
TPR
FPR
Estimation

Figure 5.1: The visualization of the binary classification metrics calculation to construct ROC curve.

The performance of the methods can be evaluated in the same sense as the binary classification since the positive (negative) class denotes the non-zero (zero) estimate of a GC connection. The binary classification metrics we used are,

- F1 score : 2TP/(2TP+FP+FN)
- False positive rate (FPR): FP/(FP+TN)
- Accuracy (ACC): (TP+TN)/(TP+TN+FP+FN)
- Matthews correlation coefficient (MCC): $\frac{\mathrm{TP}\cdot\mathrm{TN}-\mathrm{FP}\cdot\mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}}$,

where TP, TN, FP, FN are true positive, true negative, false positive, false negative respectively. The higher value of F1, ACC, MCC, TPR and lower FPR indicates the higher performance. In Figure 5.1, we visualized how the raw performance metrics were calculated from the sparse estimation methods with a single tuning parameter λ that controlled the model sparsity level. The overall performance of a method can be achieved by varying λ to get a variety of sparsity pattern, resulting in the FPR and TPR evaluate on each λ. The pair of FPR and TPR for all λ is then used for constructing the ROC (Receiver Operating Characteristic) curve. The overall performance of a method can be quantified by the area under the ROC curve (AUC). If the AUC is low, it suggested that the performance is low no matter what λ is used. Since ROC curve considered only single regularization term, we reported the overall performance metric in DGN and FGN by using the F1 score on the 2D grid of regularization pairs instead.

5.1 Ground-truth system generation

Ground-truth systems are regarded as K VAR models that the underlying GC networks coincide with the assumptions of estimation formulation as follows,

1. Common type ground truth[1]: All K models have identical topology of GC network.
2. Differential type ground truth: All K models partially shared topology of GC network and each model also has its own different pattern.
3. Fused type ground truth: same as the differential type but the common part shared VAR coefficients.

The stability of each VAR model must be ensured to prevent the divergence of the generated time-series. A VAR model with order p is stable if and only if the dynamic matrix,

$$\begin{bmatrix} A_1 & A_2 & \cdots & A_{p-1} & A_p \\ I_n & 0 & \cdots & 0 & 0 \\ 0 & I_n & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & I_n & 0 \end{bmatrix},$$

has all eigenvalue inside unit circle. In order to construct a stable VAR, we first randomized each A_r to be diagonal matrix, so that the characteristic equation of the dynamic matrix,

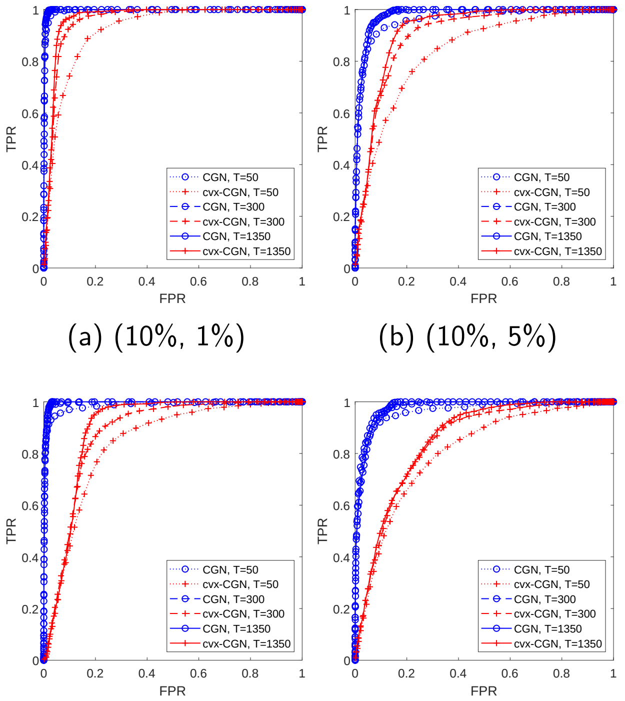
$$\prod_{i=1}^{n} (z^{p-1}(A_1)_{ii} + z^{p-2}(A_2)_{ii} + \cdots + (A_p)_{ii}) = 0,$$

[1] However, this type was not used in our experiments since we were more interested in the case of having both common and differential part

has all roots lying inside the unit circle. For $K > 1$, we repeated the generating process to obtain K stable diagonal VAR models. After the stable diagonal VAR models are constructed, we added the off-diagonal parameters in the same way as the type of ground-truth systems. If the generated models are unstable, the randomization process can be re-started until the obtained models are stable. This method is inefficient if the desired models are either too dense or having too high order, making the stability condition is difficult to be satisfied. However, dense and high-order VAR models are not of our interest.

5.2 Common GC network

The experiments were designed to illustrate a benefit of non-convex CGN over its convex counterpart and to compare them to the existing literature. The CGN formulation was compared with cvx-CGN on three factors, common density, differential density and the time-points used in the estimation. We set the penalty weight v_{ij} in (3.6) to be unity to see the performance gain only from the non-convex penalty. We compared our formulations with the existing works in the second part of this section.



(a) (10%, 1%)

(b) (10%, 5%)

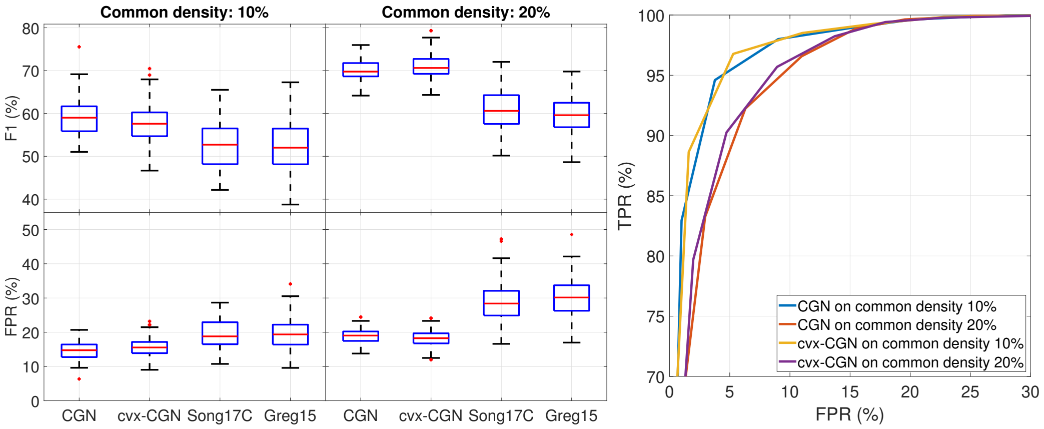
(c) (20%, 1%)

(d) (20%, 5%)

Figure 5.2: The ROC curve comparison between CGN (non-convex) and cvx-CGN (convex) without penalty weight in problem setting: (**common density**, **differential density**).

The performance comparison between CGN and cvx-CGN was evaluated in two folds. First, we varied both common and differential densities of ground-truth GC networks. This is to see the impact of the violation of both sparsity assumption from the increased common density and the violation of homogeneity assumption from the increased differential density. Second, we varied time-points to see the performance advantages of CGN over cvx-CGN. We generated 40 time-series realization of differential type joint GC networks with problem parameters as $n = 15, p = 2, K = 4$. We varied the common density to be 10% and 20% and the differential density to be 1% and 5%. The estimation parameters are $p = 2, T = 50, 300, 1350$.

The ROC curve shown in the Figure 5.2 suggested that CGN outperformed cvx-CGN in all settings. The CGN formulation has performance reduction less than cvx-CGN as the common and differential density grow. The increasing differential density has the most impact on performance in both cvx-CGN and CGN. This was an expected result since the differential density directly violated the homogeneity assumption of both cvx-CGN and CGN formulation. The time-points used in estimation significantly affect the ROC curve of cvx-CGN formulation but not significant for CGN. This result came from that sparsity recovery property has a weaker condition to be satisfied with non-convex group norm penalty than the convex group norm penalty.



(a) Distribution of F1 and FPR on common Granger networks estimations comparison.

(b) Averaged ROC curve of common Granger networks estimation.

Figure 5.3: Performance comparison between existing works and cvx-CGN and CGN formulation.

Table 5.1: Performance index for **CGN** formulation when the density of common GC networks is varied.

	Common density: 10%				Common density: 20%			
	CGN	cvx-CGN	Song17C	Greg15	CGN	cvx-CGN	Song17C	Greg15
F1	**59.2** (4.4)	57.7 (4.6)	52.4 (5.8)	52.6 (6.3)	70.0 (2.3)	**70.9** (3.0)	61.0 (4.9)	59.6 (4.4)
FPR	**14.7** (2.7)	15.7 (2.8)	19.6 (4.4)	19.4 (4.7)	19.0 (2.0)	**18.1** (2.4)	28.6 (6.0)	30.3 (5.8)
TPR	**100.0** (0.0)	**100.0** (0.0)	99.8 (0.8)	99.7 (1.0)	**100.0** (0.0)	**100.0** (0.1)	99.9 (0.5)	99.8 (0.7)
ACC	**86.7** (2.4)	85.8 (2.5)	82.3 (4.0)	82.4 (4.2)	84.4 (1.6)	**85.1** (2.0)	76.5 (4.9)	75.2 (4.7)
MCC	**60.0** (4.1)	58.6 (4.2)	53.5 (5.4)	53.7 (5.8)	66.1 (2.4)	**67.1** (3.1)	56.0 (5.5)	54.4 (5.1)

The second part concerns on the performance comparison of our methods with the literature. We compared our formulations to the following formulations,

- **Song17C:** a similar group lasso approach to cvx-CGN by Songsiri (2017) but $v_{ij} = 1$ in (3.6).
- **Greg15:** a combination of group lasso and Tikhonov regularization approach (Gregorova et al., 2015). This work was similar to Song17C but additionally penalized the diagonal part of VAR coefficients with ℓ_2 penalty. In our opinion, the diagonal part is not involved in inferring the Granger causality among the variables, so such regularization only affects the model parameter biases. The penalty parameters of both group lasso and Tikhonov regularization were set to be equal in Gregorova et al. (2015).

For generating ground-truth VAR systems, we varied the density of common GC from 10% to 20% and set the density of differential GC as 5%, to see the effect of ground-truth network density on the performances. The ground truth system parameters are $n = 20, p = 1, K = 5, T = 100$ and we also set $p = 1$ in the estimation. The performance indices were evaluated on the common part of the ground-truth network.

The boxplot in Figure 5.3(a) and the averaged metrics in Table 5.1 suggested that both CGN and cvx-CGN outperformed the other methods with significant improvement gains in the case of 20% common density. This directly came from introducing a reasonable choice of v_{ij} while setting equal v_{ij}'s in Song17C did not exploit different likelihood of zero locations in VAR parameters. The Tikhonov regularization of Greg15 has no direct effect on the GC estimation performance as the evaluation was conceptually taken only on the off-diagonal part of VAR coefficients. The effects of common density on our formulations were illustrated in Figure 5.3(b) where the performances dropped as the density increased, which is a typical characteristic in sparse-inducing framework. However, the effect on the actual performance that is also contributed from the penalty selection (Figure 5.3(a) and Table 5.1) was the opposite; the F1 score was higher when the density increased.

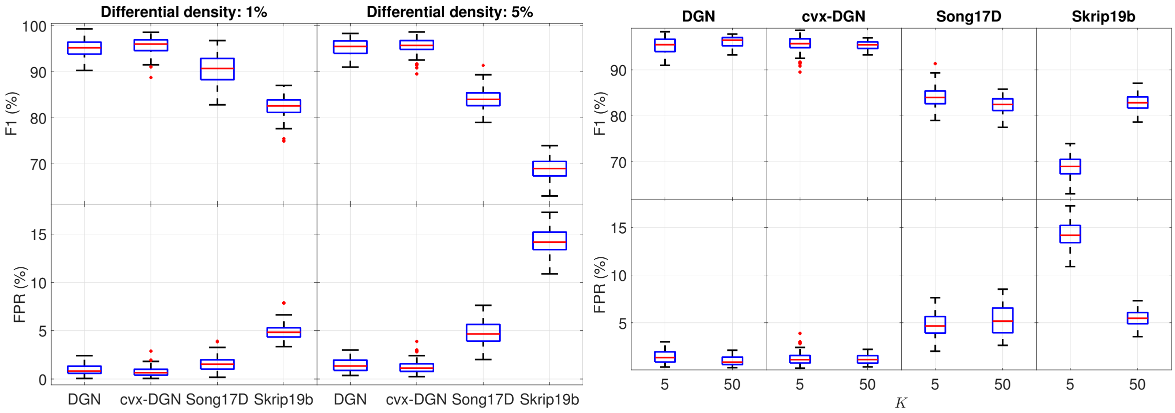
5.3 Common and differential GC network

This section concerns on the DGN and cvx-DGN performance comparison with the existing works that included the same prior knowledge on the models relation as in DGN formulation. The following is the literature aiming at decomposing GC networks in the same sense.

- **Skrip19b**: a two-stage approach (Skripnikov and Michailidis, 2019b) that estimated the parameters of the common network using a group lasso (similar to our CGN) in the first step and subsequently estimates the individual components based on the resulting common network. This approach does not guarantee a global optimal solution as the parameters were estimated in sequential steps, not being optimized in batch. They claimed that the number of models has impact on the performance; the estimation results improved with the number of models.

- **Song17D**: a group lasso combination approach (Songsiri, 2017) which is essentially the cvx-DGN but the choice of penalty weight was not specified, so it was set to unity in this experiment.

Since the ground-truth density directly affected the performance of the sparsity recovery property, we set the density of common GC to 10% and varied the density of differential GC from 1% to 5% to illustrate the effect of differential network density. For a fair comparison, in Skrip20b, they signify that their method can perform better as the number of the models (K) grows. Therefore, we also include the case of varying number of models to be $K = 5, 50$ to examine how the number of models has an impact on the performance.

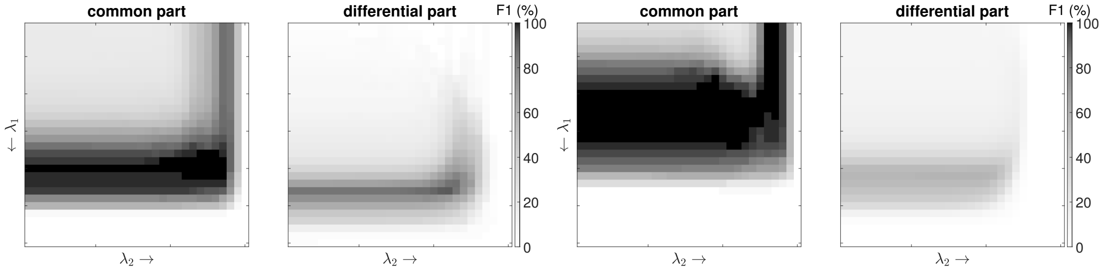


(a) Performance distribution of the estimation when differential density was **1%** and **5%**.

(b) Performance distribution of the estimation when $\mathbf{K = 5, 50}$.

Figure 5.4: Effects on estimated GC performance under the changes in density and number of models (K).

The averaged performance metrics reported in Table 5.2 has the same trends as the median of performance reported boxplot in Figure 5.4. As the differential density was increased in Figure 5.4(a), our performance reduction is less than both Song17D and Skrip19b. Song17D



(a) $K = 5$.

(b) $K = 50$.

Figure 5.5: The F1 scores of common and differential GC networks as (λ_1, λ_2) varied. The darker color indicates the higher F1 score.

Table 5.2: Averaged performance metrics of **DGN** formulation

(a) The performance index for **DGN** formulation when the density of common GC networks is varied.

	Differential density: 1%				Differential density: 5%			
	DGN	cvx-DGN	Song17D	Skrip19b	DGN	cvx-DGN	Song17D	Skrip19b
F1	95.1 (2.0)	**95.6** (1.9)	90.6 (3.1)	82.4 (2.2)	95.3 (1.8)	**95.6** (1.7)	84.1 (2.3)	68.9 (2.0)
FPR	1.0 (0.5)	**0.8** (0.5)	1.6 (0.7)	4.9 (0.8)	1.4 (0.7)	**1.2** (0.6)	4.7 (1.3)	14.3 (1.4)
TPR	98.0 (1.5)	97.5 (1.7)	94.0 (3.7)	**99.6** (0.4)	**99.0** (0.7)	98.4 (1.0)	93.7 (3.4)	98.8 (0.8)
ACC	98.9 (0.5)	**99.1** (0.4)	98.0 (0.7)	95.5 (0.7)	98.6 (0.5)	**98.7** (0.5)	95.0 (0.9)	87.5 (1.1)
MCC	94.6 (2.2)	**95.2** (2.1)	89.6 (3.4)	81.6 (2.2)	94.6 (2.0)	**94.9** (1.9)	81.9 (2.5)	66.7 (2.0)

(b) The performance index for **CGN** formulation when the number of models (K) is varied.

	DGN		cvx-DGN		Song17D		Skrip19b	
	$K = 5$	$K = 50$	$K = 5$	$K = 50$	$K = 5$	$K = 50$	$K = 5$	$K = 50$
F1	95.3 (1.8)	96.1 (1.2)	95.6 (1.7)	95.3 (0.9)	84.1 (2.3)	82.2 (1.9)	68.9 (2.0)	82.9 (1.6)
FPR	1.4 (0.7)	1.0 (0.5)	1.2 (0.6)	1.2 (0.5)	4.7 (1.3)	5.3 (1.6)	14.3 (1.4)	5.5 (0.8)
TPR	99.0 (0.7)	98.4 (0.8)	98.4 (1.0)	97.6 (1.0)	93.7 (3.4)	92.3 (4.0)	98.8 (0.8)	94.7 (0.9)
ACC	98.6 (0.5)	98.9 (0.4)	98.7 (0.5)	98.7 (0.3)	95.0 (0.9)	94.4 (0.9)	87.5 (1.1)	94.5 (0.7)
MCC	94.6 (2.0)	95.5 (1.3)	94.9 (1.9)	94.6 (1.1)	81.9 (2.5)	79.7 (1.9)	66.7 (2.0)	80.6 (1.8)

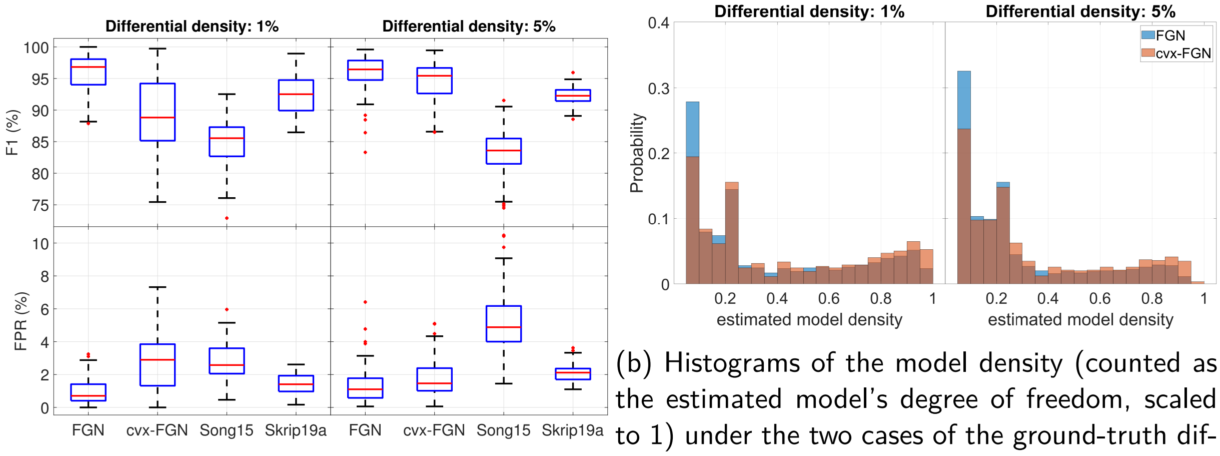
suffered from a lack of prior, so the poorer performance is expected in this formulation compared to cvx-CGN. Skrip19b may suffer from a sub-optimality of their formulation. When K increased in Figure 5.4(b), Skrip19b has considerable performance gains as they claimed, while our performances appeared not to depend on K. Although our results outperformed the others for both $K = 5$ and $K = 50$, this is a limitation of our formulations. The drawback is unclear in both CGN and cvx-CGN but it can be seen that the Song17D has a merely reduced F1 and increased FPR. This problem came from that the estimated common part, C_{ij}, is also affected by the regularization on the differential part, $B_{ij}^{(k)}$. We refer to this phenomenon as the overlapping penalization problem. This overlapping penalization is clearly presented in the F1 grid plot of all (λ_1, λ_2) pairs in Figure 5.5. From the figure, when $K = 50$, high F1 scores evaluated on the common part occurred in the different regions of (λ_1, λ_2) from those evaluated on the differential part. This issue became more severe in a higher K. In a word, the best-case performance of both the differential and common networks cannot co-exist using the same pair of (λ_1, λ_2) for large K. The choice of relative weights, v_{ij} and $w_{ij}^{(k)}$, did mitigate this issue but it did not solve the problem. Hence, achieving the best performance of both common and differential networks is not quite possible when the number of models is relatively large. However, achieving the best performance on both parts is unnecessary in practice, as we generally focus either on the common or the differential GC. In the setting that a common part is favored, we can apply the model selection to evaluate only on the common sparsity pattern, not on the total part as in our experiments. Moreover, the applications on which the differential part is focused generally involve a small K, *e.g.*, brain signals collected under various symptom stages. In such a case, the issue does not occur.

5.4 Fused and differential GC network

We demonstrated the performance of FGN and cvx-FGN over the existing works in literature. The selected works have the same prior knowledge on model relations as our formulations. The following formulations can extract the common part of the network by shrinking the differences of VAR parameters between models while the sparsity is also introduced.

- **Skrip19a**: a sparse fused-lasso approach (Skripnikov and Michailidis, 2019a) employed a combination of lasso and fused lasso to induce a sparsity on VAR coefficients and model parameter differences, respectively. The VAR sparsity obtained from their lasso term does not correspond to the characterization of GC on the all-lag VAR coefficients.
- **Song15**: a group fused-lasso approach (Songsiri, 2015) similar to cvx-FGN, except that the fused term was only taken on the consecutive models and the relative penalty weight was set to one.

The setting in this experiment is similar to Section 5.3 that the ground-truth system parameters were $n = 20, p = 1, K = 5$, the common density was fixed at 10% and differential density was varied to be 1% and 5%. The estimation parameters are $p = 1, T = 100$. The only differences that the common GC networks fixed to have same VAR coefficients for examining the performance of fused framework.



(a) Performances of GC network estimation.

(b) Histograms of the model density (counted as the estimated model's degree of freedom, scaled to 1) under the two cases of the ground-truth differential density.

Figure 5.6: Performances of FGN as the density of ground-truth differential GC networks varied.

From the boxplot in Figure 5.6(a) and the averaged metrics in Table 5.3, the FGN outperformed other methods in all settings, thanks to its formulation that accommodates the prior of having identical parameters across models. The FGN appeared to be most robust to the differential density variation compared to other methods. A decreased performance when the density increased is generally expected in sparse learning, and also was observed in the results

Table 5.3: The performance index for **FGN** formulation when the density of common GC networks is varied.

	Differential density: 1%				Differential density: 5%			
	FGN	cvx-FGN	Song15	Skrip19a	FGN	cvx-FGN	Song15	Skrip19a
F1	**95.8** (3.0)	89.2 (5.8)	85.0 (3.5)	92.5 (3.0)	**95.8** (2.9)	94.3 (3.1)	83.4 (3.4)	92.3 (1.5)
FPR	**1.0** (0.8)	2.9 (1.7)	2.8 (1.1)	1.4 (0.6)	**1.3** (1.1)	1.8 (1.2)	5.1 (1.7)	2.1 (0.5)
TPR	**99.5** (0.5)	**99.5** (0.5)	91.7 (3.7)	97.1 (2.1)	**99.4** (0.5)	98.9 (0.8)	93.8 (2.4)	97.4 (1.4)
ACC	**99.1** (0.7)	97.4 (1.5)	96.6 (1.0)	98.4 (0.6)	**98.8** (0.9)	98.3 (1.0)	94.7 (1.4)	97.8 (0.4)
MCC	**95.5** (3.2)	88.5 (5.9)	83.5 (3.8)	91.8 (3.3)	**95.2** (3.2)	93.5 (3.4)	81.1 (3.8)	91.3 (1.6)

of Song15 and Skrip19a; on the other hand, cvx-FGN performance unexpectedly increased with the differential density. This can be explained from the histogram in Figure 5.6(b) that shows empirical distributions of the estimated model's degree of freedom as (λ_1, λ_2) varied. For cvx-FGN, the portion of extracted sparse models in the 1%-density setting was less than that of 5%, leading to less number of sparse model candidates for eBIC to choose, and hence, less likely to obtain a high-performance estimated model that was supposed to be sparse. An advantage of the non-convex penalty in FGN was that the portion of extracted sparse models was higher than that of cvx-FGN. This allowed the eBIC to choose among finer choices of sparse candidates.

5.5 Improvement of non-convex formulation

The estimation error bound presented in Hu et al. (2017) depends on the ground-truth group-sparsity level. The previous results in Figures 5.3, 5.4 and 5.6 did not show significant differences between the convex and non-convex performances as the ground-truth systems were relatively sparse and possibly, the true sparsity levels stayed in the range that allowed both formulations to perform closely. The GREC as an assumption to obtain a recovery bound (Hu et al., 2017) can be prone to be violated in a low-sample-high-dimension setting. To illustrate the benefit of the non-convex penalty over the convex counterparts, we increased the ratio of variables to data samples from 4:1 to 8:1 by setting the ground-truth system parameters as $(n, p, K, T) = (20, 1, 5, 100)$ and $(20, 3, 5, 150)$. The densities of common and differential GC were set to 10% and 5%, respectively. The convergence to a global optimum for non-convex problems generally depends on the algorithm initialization. As also pointed out in Wen et al. (2018) that the improvement of non-convex over convex penalty may not be distinct for some choice of initialization (such as zero in the regression problems.) In our implementation, we started the algorithm for non-convex formulations with the least-squares solution.

Directly seen from Figure 5.7 and Table 5.4, the non-convex formulations outperformed their convex counterparts in this setting. For a fixed penalty parameter, as the non-convex formulations yield sparser solutions than the convex ones, the false positives can be much reduced when the true system is sufficiently sparse, supported by a great reduction of FPR from the non-convex models. Despite the superiority in performance of non-convex formulation, we should consider the convex formulations first if the time-points to parameters ratio is sufficiently large in the application that the uniqueness of the solution is significant.

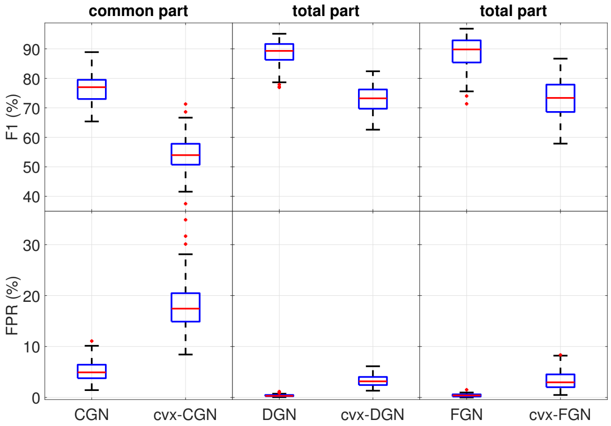


Figure 5.7: The performance advantages of non-convex penalty functions over convex penalty functions in each formulation.

Table 5.4: Performance index between of non-convex formulations (CGN, DGN and FGN) and convex formulations (cvx-CGN, cvx-DGN and cvx-FGN).

	CGN	cvx-CGN	DGN	cvx-DGN	FGN	cvx-FGN
F1	**76.6** (5.0)	54.1 (5.9)	**88.5** (4.1)	72.9 (4.7)	**88.5** (5.4)	73.3 (6.7)
FPR	**5.2** (2.0)	18.1 (4.8)	**0.4** (0.2)	3.3 (1.1)	**0.4** (0.3)	3.5 (1.9)
TPR	92.0 (6.4)	**98.7** (2.1)	**81.4** (6.7)	69.2 (7.8)	**81.9** (9.0)	70.5 (10.3)
ACC	**94.5** (1.6)	83.5 (4.2)	**97.1** (0.9)	92.8 (1.1)	**97.1** (1.2)	92.9 (1.7)
MCC	**75.2** (5.0)	54.8 (5.6)	**87.4** (4.1)	69.2 (5.0)	**87.5** (5.4)	69.8 (7.4)

5.6 Application of CGN: Supervised classification

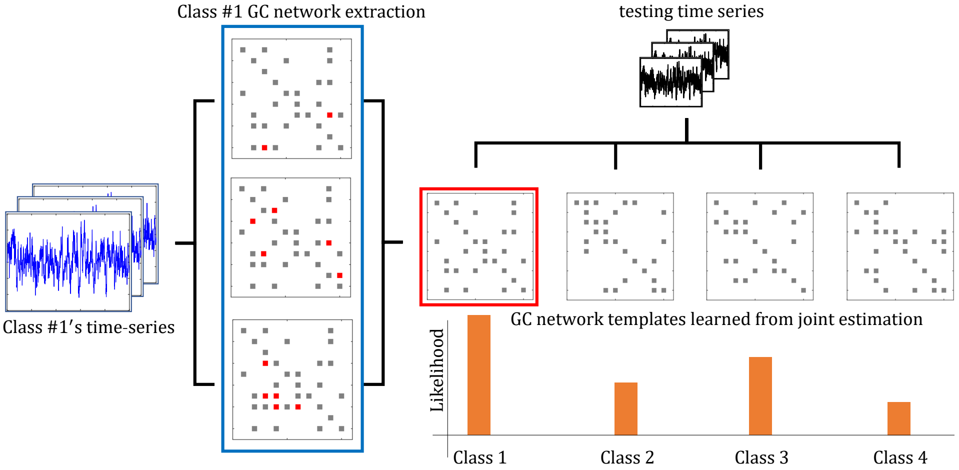

Class #1 GC network extraction
testing time series
Class #1′s time-series
GC network templates learned from joint estimation
Likelihood
Class 1
Class 2
Class 3
Class 4

Figure 5.8: The classification scheme from learned common GC network in each class.

We demonstrated a practical use of learning common GC network for multi-class classification using log-likelihood ratio test. Suppose there is one unknown class time-series to be classified into M classes and each class also has K set of time-series, the goal of this

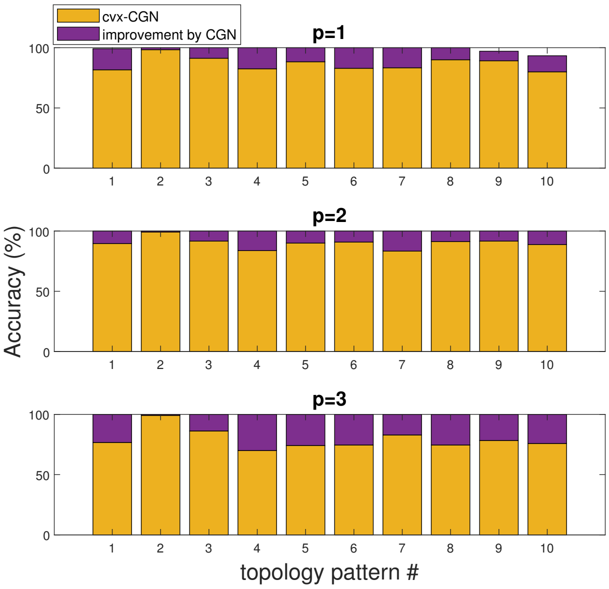


Figure 5.9: The classification accuracy improvement of each class using GC network learned from CGN (purple) over the accuracy of GC network learned from cvx-CGN (yellow) when the estimation model order was set at $p = 1, 2, 3$ when the true model order is 2.

experiment is to learn the common GC network from K set of time-series in each class[2] and then use the common GC networks to classify the unknown time-series by the likelihood ratio test. The likelihood of each class respected to the given time-series is evaluated by fitting the time-series to the VAR model with sparsity pattern of each class. The unknown time-series belongs to the class with highest likelihood as shown in Figure 5.8.

We generated 10 different topology of Granger networks to represent each class ($M = 10$). The problem parameters were $n = 15, p = 2, K = 5$, common density was set to be 10%, 20% and differential density was set to be 1%, 5%. In each topology, we estimated common network using estimation parameters as $p = 2, T = 50, 300, 1350$ for diversifying the learned common networks. We generated the testing time-series with length 50 according to each class for 20 realizations. The time-series are then fitted to the common GC network on each class with parameters $p = 1, 2, 3$ to see if the wrongly chosen model order affect the performance. In total, the number of realization in this experiment is 240.

The classification performance gain of the common network learned from CGN over cvx-CGN was presented in Figure 5.9. The classification using the common GC network learned with CGN formulation has significant accuracy improvement from the classification based on cvx-CGN. With the non-convex penalty, the classification has a near-perfect classification rate. Moreover, the accuracy improvement was still obtained even if the model order (p) was wrongly chosen.

[2] In general, the number K can be different in each class.

5.7 fMRI ADHD-200 data

Children with Attentive deficit hyperactivity disorder (ADHD) are known to suffer from abnormalities that originated in some brain regions, both functionally and structurally, when compared to typically developing children (TDC). We aim to explain differences of effective brain connectivity underlying the two groups using the ADHD-200 competition data set (Bellec et al., 2017). The fMRI time-series data were obtained from the ADHD-200 data sets by the ADHD 200 consortium and are available at: https://www.nitrc.org/plugins/mwiki/index.php/neurobureau:AthenaPipeline. We pre-processed the data using 14 steps according to the Athena functional data processing pipeline Bellec et al. (2017) but without the bandpass filtering step ($0.009 - 0.08$ Hz) since Sato et al. (2012) reported that ADHD and TDC (control) groups were highly discriminative when using the cross-spectral density at the frequency around 0.2 Hz as a feature. The data were collected from the NYU site and screened under the criteria: i) the subjects were male adolescents of 7-17 years old (11.71 ± 3.11), ii) the ADHD and TDC groups were age-matched, iii) the subjects had no secondary diagnosis, iv) the subjects are right-handed with a score larger than 0.1, where the score ranged from -1 to 1 (from left-handed to right-handed), iv) the subjects had verbal IQ in the range of 98-112, and v) ADHD subjects were combined subtype, *i.e.*, ADHD type with the presence of both inattention and hyperactivity/impulsivity. Under these selections, we obtained 18 subjects for each of the ADHD and TDC groups. The resting-state fMRI time series were averaged over voxels within AAL-atlas (Tzourio-Mazoyer et al., 2002) regions of interest (ROIs) as shown in Table 5.6, resulting in 116-channel time series with time points of 172.

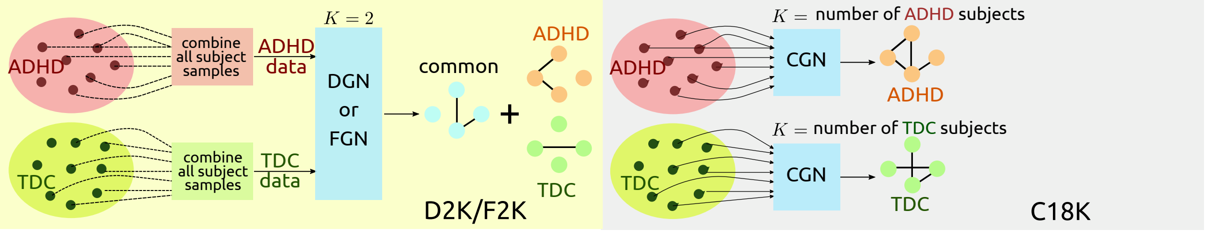

ADHD
combine all subject samples
ADHD data
$K = 2$
DGN or FGN
common
ADHD
TDC
combine all subject samples
TDC data
TDC
D2K/F2K
$K =$ number of ADHD subjects
ADHD
CGN
ADHD
$K =$ number of TDC subjects
TDC
CGN
TDC
C18K

Figure 5.10: The D2K, F2K, and C18K schemes of learning ADHD and TDC networks.

As the number of samples is sufficiently moderate compared to the number of variables, the distinction between the non-convex and convex formulations may not be significant. For this reason, we used only the convex formulations to avoid the local optimum, or the algorithm convergence issues. We applied our cvx-CGN, cvx-DGN, and cvx-FGN in different setup as shown in Figure 5.10 called C18K, F2K and D2K which are,

1. **D2K:** For each of TDC and ADHD groups, we pooled data from all subjects, so the sample sizes of each group increased. Combined data from two groups are the input to cvx-DGN with $K = 2$ and the outputs are ADHD and TDC networks with extracted common and differential parts.

2. **F2K:** The scheme was the same as D2K but cvx-FGN was used to estimate the two networks.
3. **C18K:** We applied cvx-CGN to learn a common network among 18 subjects ($K = 18$) on each of ADHD and TDC data sets.

All schemes resulted in two estimated GC networks, each for ADHD and TDC subjects. The networks obtained from D2K/F2K contain a common structure across two groups, and the individual differences explained the characteristics of each group, while the networks from C2K represent dominant connections that are common across subjects in each group. As the number of AAL ROIs is 116, it is quite complicated to visualize the results as a graphical model. We further analyzed brain connections in the estimated GC network using the *edge betweenness centrality* measure (Rubinov and Sporns, 2010) used in network theory, which is the number of times that an edge of interest appears in all existing shortest paths between any two nodes of a graph. As input to compute the score, the weights of all edges in the graph are required.

In GC context, a higher value of GC indicates a stronger connection between two regions, so we used the reciprocal of GC to be the weight in the graph; specifically, $1/\|B_{ij}^{(k)}\|_2$ can be used as a proxy-distance between the regions i and j since a stronger GC connection should indicate a shorter path. If the difference of edge centrality between ADHD and TDC networks is significantly high, the percentage that such GC edge in ADHD network appears in the shortest paths of the graph has changed notably from the TDC network (*i.e.*, loosely speaking, such brain connection in the ADHD network is dominantly different from TDC.) We divided the centrality differences into two types, the missing and extra types, in which the centrality measure of ADHD is lower than that of TDC and vice versa. Brain connections corresponding to the three highest centrality differences were presented in Table 5.5.

From Table 5.5, the differences between ADHD and TDC were primarily concentrated on two regions: the *orbitofrontal region (ORB)*, and those associated with the *limbic system*. The orbitofrontal region is known to associated with a reward-motivation system that responds to a reward or punishment (Rolls et al., 2020). Two ROIs that are part of the limbic system and found in this study were i) the anterior cingulate cortex (ACG), which was related to emotion (Bush et al., 2000), decision making and social interaction (Lavin et al., 2013), and ii) the parahippocampal gyrus (PHG) that involved memory retrieval and emotion processing (Aminoff et al., 2013). Among those two regions, three brain connections having significant differrences of centrality scores between the two networks are shown in Figure 5.11(a).

First, the link no.7 connecting from *the left superior frontal gyrus (ORBsupmed) to the left ACG* were missing in ADHD. The two areas are believed to explain anti-social behavior of ADHD patients, supported by anatomical evidence that subjects with focal brain damages in those areas also exhibited this behavior (Bechara, 2004), and by a decrease of functional brain connectivity between the two regions in the subjects with social anxiety disorder (Hahn et al.,

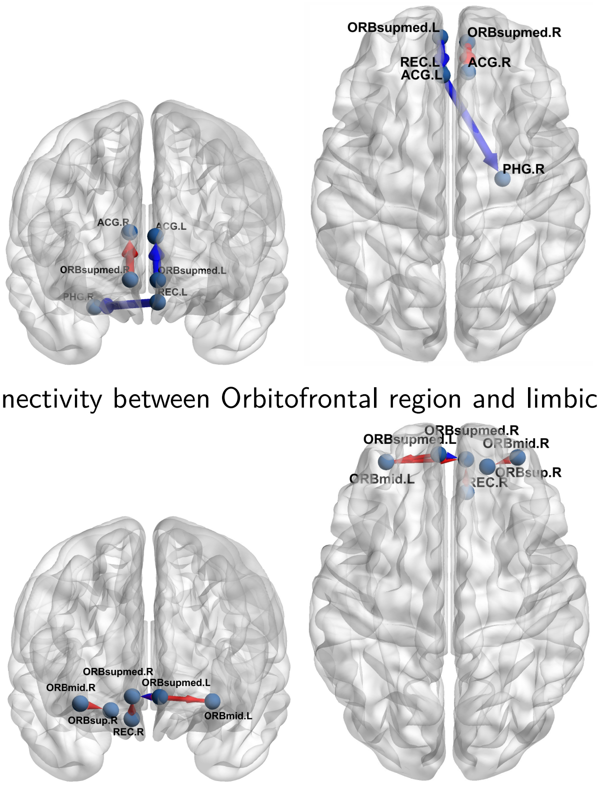
Table 5.5: Distinct connections between ADHD and TDC, ranked by the three highest absolute differences of the edge betweenness centrality.

No.	Scheme	Cause	Effect	Centrality difference	Associated system
ADHD < TDC (missing)					
1	D2K	Anterior cingulate gyrus L	Anterior cingulate gyrus R	-648	Limbic system
2		Anterior cingulate gyrus R	Fusiform gyrus R	-474	Limbic-Temporal
3		Cerebellum 3 L	Anterior cingulate gyrus L	-442	Cerebellar-Limbic
4	F2K	Rectus gyrus L	Parahippocampal gyrus R	-282	Orbitofrontal-Limbic
5		Amygdala L	Superior temporal gyrus L	-141	Limbic-Temporal
6		Parahippocampal gyrus R	Inferior frontal gyrus (orbital) R	-130	Limbic-Frontal
7	C18K	Superior frontal gyrus (medial orbital) L	Anterior cingulate gyrus L	-349	Orbitofrontal-Limbic
8		Superior frontal gyrus (medial orbital) L	Superior frontal gyrus (medial orbital) R	-125	Orbitofrontal
9		Rolandic operculum R	Precentral gyrus R	-124	
ADHD > TDC (extra)					
10	D2K	Superior frontal gyrus (medial orbital) R	Anterior cingulate gyrus R	367	Orbitofrontal-Limbic
11		Olfactory cortex L	Insula L	344	Olfactory-insular
12		Temporal pole (superior) L	Putamen L	327	Temporal-Frontal
13	F2K	Middle frontal gyrus (orbital) R	Superior frontal gyrus (orbital) R	164	Orbitofrontal
14		Temporal pole (superior) L	Rolandic operculum L	164	Temporal-Operculum
15		Rectus gyrus R	Superior frontal gyrus (medial orbital) R	143	Orbitofrontal
16	C18K	Supplementary motor area R	Precuneus R	306	
17		Superior frontal gyrus (medial orbital) L	Middle frontal gyrus (orbital) L	296	Orbitofrontal
18		Middle frontal gyrus (orbital) L	Superior frontal gyrus (medial orbital) R	282	Orbitofrontal

* L/R denotes the left or right hemisphere. ** The regions with (orbital) or (medial orbital) are orbitofrontal area.
*** All the missing-type links were not in the ADHD network, except no.3. All the extra-type links were not in the TDC network, except no.18.

2011). Second, not only that we found this connection missing in the left hemisphere, but the extra connection in the *right hemisphere* of the ADHD network was also discovered. This result was supported by Tomasi and Volkow (2012) that found higher functional connectivity between ORB and ACG in ADHD. Finding the two connections convinced us that the increased centrality score in the right hemisphere of ADHD network may account for a reward-motivation dysfunction and the decreased score in the left hemisphere may explain anti-social behavior. Third, the connection no.4 from the *left rectus gyrus (REC) to the right PHG* was missing in the ADHD network, implying a broken connection from the ORB region to part of the limbic system. This can be partly supported by Itani et al. (2019) that features extracted from the PHG region was highly discriminative for ADHD classification using decision trees. In addition, Sung et al. (2016) discovered that subjects with REC resection had an impairment of memory recall and language skills when tested with the mini-mental state examination, so the REC may involve with the limbic functions, agreeing with our third connection that was missing.

The five links in Figure 5.11(b) associated with the system of orbitofrontal (no. 8,13,15,17,18 in Table 5.5) shows several connections *within the ORB region* that had distinct score differences between ADHD and TDC. For ADHD patients, the ORB region was responsible for the reward learning sensitivity or a slower learning rate when the objective of the reward-related task was changed (Itami and Uno, 2002). Out of the five connections, a common cause of the connections no.8 and 17 was the left ORBsupmed, the region from which the extracted feature was effective for ADHD classification (Itani et al., 2019). The extra connection (no.14) from *the right REC to the right ORBsupmed* in ADHD agreed with Tang et al. (2020) that such functional link was significant for ADHD classification using linear discriminant analysis. In addition, the REC and ORBsupmed regions were focused in a reward-system dysfunction study for obese groups or binge eating disorder (Shott et al.,



(a) Connectivity between Orbitofrontal region and limbic system

(b) Connectivity within orbitofrontal region

Figure 5.11: The coronal and axial view of the selected brain connections among the orbitofrontal region and part of limbic system that have distinct centrality differences between the ADHD and TDC networks. The blue (red) directed edges show a missing (extra) connections in ADHD.

2015), symptoms that were related to ADHD as reported in Seymour et al. (2015).

We can draw two conclusions regarding how the three formulations infer different brain connections shown in Table 5.5. First, most links obtained from C18K are concentrated within the ORB region, while the ROIs involved connections from D2K/F2K can be more diverse. In C18K setting, the ADHD and TDC networks were estimated separately; each revealed significant characteristics that were common within each group. The ORB possibly explains a brain functioning for both groups, but with different degrees for ORB subregions, so the centrality differences were found concentrated in this region. On the contrary, the F2K/ D2K formulations jointly estimated the two networks where the differential part was freely encouraged to present in any regions, so the network differences between the two groups occurred in several areas.

Second, despite the same paradigm in both F2K and D2K settings, the magnitudes of centrality differences of D2K were in a higher scale than those of F2K. This can be explained from the penalty being used since the similarity of parameters across models affects the weight

Table 5.6: AAL atlas

#	ROI	#	ROI (continue)	#	ROI (continue)
1	Precentral gyrus, Left	40	Parahippocampal gyrus, Right	79	Heschl gyrus, Left
2	Precentral gyrus, Right	41	Amygdala, Left	80	Heschl gyrus, Right
3	Superior frontal gyrus (dorsolateral), Left	42	Amygdala, Right	81	Superior temporal gyrus, Left
4	Superior frontal gyrus (dorsolateral), Right	43	Calcarine cortex, Left	82	Superior temporal gyrus, Right
5	Superior frontal gyrus (orbital), Left	44	Calcarine cortex, Right	83	Temporal pole (superior), Left
6	Superior frontal gyrus (orbital), Right	45	Cuneus, Left	84	Temporal pole (superior), Right
7	Middle frontal gyrus, Left	46	Cuneus, Right	85	Middle temporal gyrus, Left
8	Middle frontal gyrus, Right	47	Lingual gyrus, Left	86	Middle temporal gyrus, Right
9	Middle frontal gyrus (orbital), Left	48	Lingual gyrus, Right	87	Temporal pole (middle), Left
10	Middle frontal gyrus (orbital), Right	49	Superior occipital gyrus, Left	88	Temporal pole (middle), Right
11	Inferior frontal gyrus (opercular), Left	50	Superior occipital gyrus, Right	89	Inferior temporal gyrus, Left
12	Inferior frontal gyrus (opercular), Right	51	Middle occipital gyrus, Left	90	Inferior temporal gyrus, Right
13	Inferior frontal gyrus (triangular), Left	52	Middle occipital gyrus, Right	91	Cerebellum Crus1, Left
14	Inferior frontal gyrus (triangular), Right	53	Inferior occipital gyrus, Left	92	Cerebellum Crus1, Right
15	Inferior frontal gyrus (orbital), Left	54	Inferior occipital gyrus, Right	93	Cerebellum Crus2, Left
16	Inferior frontal gyrus (orbital), Right	55	Fusiform gyrus, Left	94	Cerebellum Crus2, Right
17	Rolandic operculum, Left	56	Fusiform gyrus, Right	95	Cerebellum 3, Left
18	Rolandic operculum, Right	57	Postcentral gyrus, Left	96	Cerebellum 3, Right
19	Supplementary motor area, Left	58	Postcentral gyrus, Right	97	Cerebellum 4_5, Left
20	Supplementary motor area, Right	59	Superior parietal gyrus, Left	98	Cerebellum 4_5, Right
21	Olfactory cortex, Left	60	Superior parietal gyrus, Right	99	Cerebellum 6, Left
22	Olfactory cortex, Right	61	Inferior parietal gyrus, Left	100	Cerebellum 6, Right
23	Superior frontal gyrus (medial), Left	62	Inferior parietal gyrus, Right	101	Cerebellum 7b, Left
24	Superior frontal gyrus (medial), Right	63	Supramarginal gyrus, Left	102	Cerebellum 7b, Right
25	Superior frontal gyrus (medial orbital), Left	64	Supramarginal gyrus, Right	103	Cerebellum 8, Left
26	Superior frontal gyrus (medial orbital), Right	65	Angular gyrus, Left	104	Cerebellum 8, Right
27	Rectus gyrus, Left	66	Angular gyrus, Right	105	Cerebellum 9, Left
28	Rectus gyrus, Right	67	Precuneus, Left	106	Cerebellum 9, Right
29	Insula, Left	68	Precuneus, Right	107	Cerebellum 10, Left
30	Insula, Right	69	Paracentral lobule, Left	108	Cerebellum 10, Right
31	Anterior cingulate gyrus, Left	70	Paracentral lobule, Right	109	Vermis 1_2
32	Anterior cingulate gyrus, Right	71	Caudate, Left	110	Vermis 3
33	Median cingulate gyrus, Left	72	Caudate, Right	111	Vermis 4_5
34	Median cingulate gyrus, Right	73	Putamen, Left	112	Vermis 6
35	Posterior cingulate gyrus, Left	74	Putamen, Right	113	Vermis 7
36	Posterior cingulate gyrus, Right	75	Pallidum, Left	114	Vermis 8
37	Hippocampus, Left	76	Pallidum, Right	115	Vermis 9
38	Hippocampus, Right	77	Thalamus, Left	116	Vermis 10
39	Parahippocampal gyrus, Left	78	Thalamus, Right		

of GC networks. F2K used the fused lasso to encourage a parameter similarity between ADHD and TDC models, resulting in small centrality differences between the two groups. In contrast, D2K used the group lasso to enforce a differential structure of each individual network separately, so the individual model parameters can be shrunk in different degrees, resulting in a larger scale of centrality differences. This conclusion suggests that the centrality difference ranking should be made on each formulation separately since the scores are on a different scale.

Chapter VI

CONCLUSION AND DISCUSSION

This thesis aimed to extend the joint Granger graphical model estimation by combining the strength of existing regularization techniques presented in multiple literature. We proposed three sparse formulations namely CGN, DGN, and FGN in Chapter 3 for estimating multiple Granger causality networks with common causality structure across multiple time series and differential structures belonging to individual time series. These formulations can be applied to brain connectivity analysis where we are interested in a group-level inference and connectivity differences among subject conditions. The proposed formulations employed the group and fused lasso penalties with a penalty weight for enhancing the accuracy of the estimation. The non-convex $\ell_{2,1/2}$ penalty was used to further improve the estimation in low-sample settings. The estimation problems were used in combination with the extended BIC as a model selection criterion which selected an optimal pair of penalty parameters, thus completing our scheme of learning multiple GC networks at optimal sparsity.

The effectiveness of the formulations was demonstrated in Chapter 5. On average, our approaches improved F1 and FPR by 3-26% and 0.6-13%, respectively, over existing *sparse multiple Granger graphical model* methods in the literature. The main factor that determined CGN's accuracy was the density of the common ground-truth network, while DGN/FGN were slightly affected by the density of the differential ground-truth network. Contrary to previous results, DGN/FGN's accuracy was favorably insensitive to the number of models (K), and their performance improved relative to earlier methods even when K was small. However, note that the number of variables grows linearly as K increases, thus affecting the computational complexity from an algorithm's point of view.

Not without drawbacks, our framework suffers from multiple weaknesses. We provide a discussion on each drawback and a solution in the following.

Varying penalty parameters The causality learning scheme presented in Figure 3.5 employed the proposed formulations to estimate models with various degrees of sparsity through adjusting (λ_1, λ_2). For sparse formulations that have a single penalty parameter (such as lasso, group lasso, or fused lasso), it is possible to derive a range of the penalty parameters in closed-form, ordered by the model sparsity they induce from densest to sparsest. This range generally depends on the sample size and problem data. Unfortunately, for sparse-inducing problems with two or more penalties, it is difficult to derive such a range analytically. Due to this limitation, a heuristic approach is needed to create a range for (λ_1, λ_2) by setting the upper bound of λ_1 to its critical value as if there was only λ_1 and the same of λ_2.

Algorithms of non-convex formulation Convergence to a global optimum for non-convex problems generally depends on the algorithm initialization. As also pointed out in Wen et al. (2018), non-convex penalties may not show any improvements or even distinctions over convex penalties for some choices of initialization (such as zero in the regression problems.) In our implementation, the algorithm of non-convex formulations started with the least-squares solution. When solving the problem with a series of (λ_1, λ_2), a common remedy is to use the solution associated with the previous pair to initiate the algorithm.

Overlapped penalization DGN's penalty consists of two terms that penalize some overlapping groups of parameters. As such, the estimated common part, C_{ij}, is also affected by the regularization of the differential part, $B_{ij}^{(k)}$. For a large K, as we varied the pair (λ_1, λ_2) on a grid range, the best solution (in terms of highest F1) evaluated on the common network can be much different from the one evaluated on the differential network; see more experimental results in the supplementary material. In other words, the separate best-case performances of the differential and common networks cannot co-exist using the same pair of (λ_1, λ_2) for large K. The choice of relative weights, v_{ij} and $w_{ij}^{(k)}$, partly mitigates this issue but it does not completely solve the problem. However, achieving the best performance on both the common and differential parts at the same time may not be necessary in practice, since we generally focus either on the common or the differential GC when analyzing results. In settings where the common GCs are more informative, we can select a model that benefits evaluating the common sparsity pattern, and not worry about the total GC network as in our experiments. Moreover, situations where the differential GC is of more interest generally involve a small K (*e.g.*, brain signals collected under various symptom stages), and in cases of small K, this issue does not occur.

We also provided two applications of the proposed methods. First, we applied CGN and cvx-CGN to extract the common GC networks of many different groups. The sparsity pattern of each common network was used in the constrained least-square VAR estimation from a given time series. The time series were classified into a group with the highest likelihood from the constrained least-square. The results yielded that CGN outperformed cvx-CGN even when the order of the VAR model was wrongly chosen. In the second application, the proposed methods were used to analyze the differences of effective brain connectivity (in the GC sense) between ADHD and TDC subjects with resting-state fMRI time-series data obtained from the ADHD-200 dataset. Our formulations found results that were consistent with previous studies supported by both clinical and functional evidence from ADHD literature, asserting that the orbitofrontal and limbic system regions of the brain appeared highly related to ADHD.

REFERENCES

Alaíz, C. M., Barbero, A., and Dorronsoro, J. R. (2013). Group fused lasso. In *Artificial Neural Networks and Machine Learning – ICANN 2013*, pages 66–73, Berlin, Heidelberg. Springer Berlin Heidelberg.

Aminoff, E. M., Kveraga, K., and Bar, M. (2013). The role of the parahippocampal cortex in cognition. *Trends in Cognitive Sciences*, 17(8):379–390.

Attouch, H., Bolte, J., and Svaiter, B. (2013). Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized Gauss–Seidel methods. *Mathematical Programming*, 137:91–129.

Bechara, A. (2004). Disturbances of emotion regulation after focal brain lesions. volume 62 of *International Review of Neurobiology*, pages 159–193. Academic Press.

Bellec, P., Chu, C., Chouinard-Decorte, F., Benhajali, Y., Margulies, D. S., and Craddock, R. C. (2017). The Neuro Bureau ADHD-200 preprocessed repository. *NeuroImage*, 144:275–286. Data Sharing Part II.

Bolte, J., Daniilidis, A., Ley, O., and Mazet, L. (2009). Characterizations of łojasiewicz inequalities: Subgradient flows, talweg, convexity. *Transactions of The American Mathematical Society*, 362:3319–3363.

Bore, J. C., Li, P., Harmah, D. J., Li, F., Yao, D., and Xu, P. (2020). Directed EEG neural network analysis by LAPPS ($p \leq 1$) penalized sparse Granger approach. *Neural Networks*, 124:213–222.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundation and trends in machine learning*, 3(1):1–122.

Bush, G., Luu, P., and Posner, M. I. (2000). Cognitive and emotional influences in anterior cingulate cortex. *Trends in Cognitive Sciences*, 4(6).

Chen, J. and Chen, Z. (2008). Extended Bayesian information critera for model selection with large model spaces. *Biometrika*, 95:759–771.

Chun, H., Zhang, X., and Zhao, H. (2015). Gene regulation network inference with joint sparse Gaussian graphical models. *Journal of Computational and Graphical Statistics*, 24(4): 954–974.

Combettes, P. L. and Pesquet, J. C. (2011). *Proximal Splitting Methods in Signal Processing*, pages 185–212. Springer New York, New York, NY.

Danaher, P., Wang, P., and Witten, D. M. (2014). 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.

de Abril, I. M., Yoshimoto, J., and Doya, K. (2018). Connectivity inference from neural recording data: Challenges, mathematical bases and research directions. *Neural Networks*, 102:120–137.

Feng, X., Yan, S., and Wu, C. (2020). The $\ell_{2,q}$ regularized group sparse optimization: Lower bound theory, recovery bound and algorithms. *Applied and Computational Harmonic Analysis*, 49(2):381–414.

Friedman, J., Hastie, T., and Tibshirani, R. (2007). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441.

Fujita, A., Sato, J. R., Garay-Malpartida, H. M., Yamaguchi, R., Miyano, S., Sogayar, M. C., and Ferreira, C. E. (2007). Modeling gene expression regulatory networks with the sparse vector autoregressive model. *BMC Systems Biology*, 1(39).

Gregorova, M., Kalousis, A., and Marchand-Maillet, S. (2015). Learning coherent Granger-causality in panel vector autoregressive models. pages 1–4, Princeton . 2015-07. Proceedings of the Demand Forecasting Workshop of the 32nd International Conference on Machine Learning.

Gu, B., Wang, D., Huo, Z., and Huang, H. (2018). Inexact proximal gradient methods for non-convex and non-smooth optimization. In *AAAI Conference on Artificial Intelligence*.

Guo, J., Levina, E., Michailidis, G., and Zhu, J. (2011). Joint estimation of multiple graphical models. *Biometrika*, 98(1):1–15.

Hahn, A., Stein, P., Windischberger, C., Weissenbacher, A., Spindelegger, C., Moser, E., Kasper, S., and Lanzenberger, R. (2011). Reduced resting-state functional connectivity between amygdala and orbitofrontal cortex in social anxiety disorder. *NeuroImage*, 56(3):881–889.

Hara, S. and Washio, T. (2013). Learning a common substructure of multiple graphical Gaussian models. *Neural Networks*, 38:23–38.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

Hu, Y., Li, C., Meng, K., Qin, J., and Yang, X. (2017). Group sparse optimization via $\ell_{p,q}$ regularization. *Journal of Machine Learning Research*, 18(30):1–52.

Huang, F. and Chen, S. (2014). Joint learning of multiple sparse matrix Gaussian graphical models. *IEEE Transactions on Neural Networks and Learning Systems*, 26(11):2606–2620.

Huang, F., Chen, S., and Huang, S. (2018). Joint estimation of multiple conditional Gaussian graphical models. *IEEE Transactions on Neural Networks and Learning Systems*, 29(7):3034–3046.

Itami, A. and Uno, H. (2002). Orbitofrontal cortex dysfunction in attention-deficit hyperactivity disorder revealed by reversal and extinction tasks. *NeuroReport*, 13(8).

Itani, S., Rossignol, M., Lecron, F., and Fortemps, P. (2019). Towards interpretable machine learning models for diagnosis aid: A case study on attention deficit/hyperactivity disorder. *PLoS ONE*, 14(4):1–20.

Lavin, C., Melis, C., Mikulan, E., Gelormini, C., Huepe, D., and Ibanez, A. (2013). The anterior cingulate cortex: an integrative hub for human socially-driven interactions. *Frontiers in Neuroscience*, 7:64.

Li, G. and Pong, T. K. (2015). Global convergence of splitting methods for nonconvex composite optimization. *SIAM Journal on Optimization*, 25(4):2434–2460.

Li, H. and Lin, Z. (2015). Accelerated proximal gradient methods for nonconvex programming. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28*, pages 379–387. Curran Associates, Inc.

Liang, X., Connelly, A., and Calamante, F. (2016). A novel joint sparse partial correlation method for estimating group functional networks. *Human Brain Mapping*, 37(3): 1162–1177.

Lozano, A. C., Abe, N., Liu, Y., and Rosset, S. (2009). Grouped graphical Granger modeling for gene expression regulatory networks discovery. *Bioinformatics*, 25(12):i110–i118.

Lütkepohl, H. (2005). *New Introduction to Multiple Time Series Analysis*. Springer.

Ma, J. and Michailidis, G. (2016). Joint structural estimation of multiple graphical models. *Journal of Machine Learning Research*, 17(166):1–48.

Manomaisaowapak, P. and Songsiri, J. (2020). Learning a common Granger causality network using a non-convex regularization. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1160–1164.

Meinshausen, N. and Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473.

Mordukhovich, B. S. and Nam, N. M. (2013). *An Easy Path to Convex Analysis and Applications*. Morgan & Claypool Publishers, 1st edition.

Parikh, N. and Boyd, S. (2014). Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):127–239.

Qiao, L., Zhang, B., Su, J., and Lu, X. (2016). Linearized alternating direction method of multipliers for constrained nonconvex regularized optimization. In Durrant, R. J. and Kim, K.-E., editors, *Proceedings of The 8th Asian Conference on Machine Learning*, volume 63 of *Proceedings of Machine Learning Research*, pages 97–109, The University of Waikato, Hamilton, New Zealand. PMLR.

Rockafellar, R. T. and Wets, R. J.-B. (1998). *Variational Analysis*. Springer Verlag, Heidelberg, Berlin, New York.

Rolls, E. T., Cheng, W., and Feng, J. (2020). The orbitofrontal cortex: reward, emotion and depression. *Brain Communications*, 2(2).

Rubinov, M. and Sporns, O. (2010). Complex network measures of brain connectivity: Uses and interpretations. *NeuroImage*, 52(3):1059–1069. Computational Models of the Brain.

Sabach, S. and Teboulle, M. (2019). Chapter 10 - Lagrangian methods for composite optimization. In *Processing, Analyzing and Learning of Images, Shapes, and Forms: Part 2*, volume 20 of *Handbook of Numerical Analysis*, pages 401 – 436. Elsevier.

Saegusa, T. and Shojaie, A. (2016). Joint estimation of precision matrices in heterogeneous populations. *Electronic Journal of Statistics*, 10(1):1341 – 1392.

Sato, J. R., Hoexter, M. Q., Castellanos, X. F., and Rohde, A. L. (2012). Abnormal brain connectivity patterns in adults with ADHD: A coherence study. *PLoS One*, 7(9):1–9.

Seymour, K. E., Reinblatt, S. P., Benson, L., and Carnell, S. (2015). Overlapping neurobehavioral circuits in ADHD, obesity, and binge eating: evidence from neuroimaging research. *CNS Spectrums*, 20(4):401–411.

Shojaie, A. and Michailidis, G. (2010). Discovering graphical Granger causality using the truncating Lasso penalty. *Bioinformatics*, 26(18):i517–i523.

Shott, M. E., Cornier, M.-A., Mittal, V. A., Pryor, T. L., Orr, J. M., Brown, M. S., and Frank, G. K. W. (2015). Orbitofrontal cortex volume and brain reward response in obesity. *International Journal of Obesity*, 39(2):214–221.

Skripnikov, A. and Michailidis, G. (2019a). Joint estimation of multiple network Granger causal models. *Econometrics and Statistics*, 10:120–133.

Skripnikov, A. and Michailidis, G. (2019b). Regularized joint estimation of related vector autoregressive models. *Computational Statistics & Data Analysis*, 139:164–177.

Songsiri, J. (2015). Learning multiple Granger graphical models via group fused lasso. In *Proceedings of the IEEE 10th Asian Control Conference (ASCC)*.

Songsiri, J. (2017). Estimations in learning Granger graphical models with application to fMRI time series. Technical report, Department of Electrical engineering, Chulalongkorn University.

Sung, J. M., Sun, P. D., Taek, M. C., Il, C. Y., Woo, S. S., and Gee, R. H. (2016). Relationship between Gyrus Rectus resection and cognitive impairment after surgery for ruptured anterior communicating artery Aneurysms. *Journal of Cerebrovascular and Endovascular Neurosurgery*, 18(3):223–228.

Tang, Y., Li, X., Chen, Y., Zhong, Y., Jiang, A., and Wang, C. (2020). High-accuracy classification of attention deficit hyperactivity disorder with l2,1-norm linear discriminant analysis and binary hypothesis testing. *IEEE Access*, 8:56228–56237.

Tao, Q., Huang, X., Wang, S., Xi, X., and Li, L. (2016). Multiple Gaussian graphical estimation with jointly sparse penalty. *Signal Processing*, 128:88–97.

Themelis, A. and Patrinos, P. (2020). Douglas–Rachford splitting and ADMM for nonconvex optimization: Tight convergence results. *SIAM Journal on Optimization*, 30(1): 149–181.

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

Tomasi, D. and Volkow, N. D. (2012). Abnormal functional connectivity in children with attention-deficit/hyperactivity disorder. *Biological Psychiatry*, 71(5):443–450.

Tzourio-Mazoyer, N., Landeau, B., Papathanassiou, D., Crivello, F., Etard, O., Delcroix, N., Mazoyer, B., and Joliot, M. (2002). Automated anatomical labeling of activations in SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain. *NeuroImage*, 15(1):273–289.

Wang, F., Cao, W., and Xu, Z. (2018). Convergence of multi-block bregman ADMM for nonconvex composite problems. *Science China Information Sciences*, 61(122101).

Wang, H. and Leng, C. (2008). A note on adaptive group lasso. *Computational Statistics & Data Analysis*, 52(12):5277–5286.

Wang, Y., Yin, W., and Zeng, J. (2019). Global convergence of ADMM in nonconvex nonsmooth optimization. *Journal of Scientific Computing*, 78(1):29–63.

Wen, F., Chu, L., Liu, P., and Qiu, R. C. (2018). A survey on nonconvex regularization-based sparse and low-rank recovery in signal processing, statistics, and machine learning. *IEEE Access*, 6:69883–69906.

Wilms, I., Barbaglia, L., and Croux, C. (2018). Multiclass vector auto-regressive models for multistore sales data. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 67(2):435–452.

Xu, Y., Liu, M., Lin, Q., and Yang, T. (2017a). ADMM without a fixed penalty parameter: Faster convergence with new adaptive penalization. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30, pages 1–11.

Xu, Z., Figueiredo, M., and Goldstein, T. (2017b). Adaptive ADMM with spectral penalty parameter selection. In Singh, A. and Zhu, J., editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 718–727. PMLR.

Yao, Q., Kwok, J. T., Gao, F., Chen, W., and Liu, T. (2017). Efficient inexact proximal gradient algorithm for nonconvex problems. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, page 3308–3314, Melbourne, Australia. AAAI Press.

Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of The Royal Statistical Society, Series B*, 68:49–67.

Yuan, Y., Soh, D. W., Yang, X., Guo, K., and Quek, T. Q. S. (2021). Joint network topology inference via structured fusion regularization.

Zhang, S., Qian, H., and Gong, X. (2016). An alternating proximal splitting method with global convergence for nonconvex structured sparsity optimization. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, page 2330–2336, Phoenix, Arizona. AAAI Press.

Appendix A

ADDITIONAL EXPERIMENTAL RESULTS

We provided the experiments that we used to tune the algorithm parameters in the following.

Experiment: Initial point selection for nmAPG algorithm.

Objective The non-convex problems are known to have multiple local-minima and heavily affected by the choices of initialization. In these experiments, we aim to select an initial point for solving the regularized least-square problem using non-convex group norm penalty. One of the heuristic approaches was initializing the non-convex problem with the solution of group lasso regression, but we aim to find if there are other easier choices than solving an optimization problem.

Setting In this experiment, we investigated the initialization in non-convex group norm penalty in a simple linear regression model. The ground-truth model is

$$b = G\tilde{x} + \epsilon,$$

where $b \in \mathbf{R}^{200}, \tilde{x} \in \mathbf{R}^{1000}$ with SNR of 20dB. We also assumed that $\tilde{x}$ is block-sparse with block size 10 and there are 10 non-zero groups out of 100 groups.

We considered 7 initialization in comparison which are

- $x_{\text{zero}} = \mathbf{0}$, (zero initialization)
- $x_{\text{ridge}} = (G^T G + 0.1 I_{1000})^{-1} G^T b$, (ridge solution initialization, $\lambda = 0.1$)
- $x_{\text{minnorm}} = G(GG^T)^{-1} b$, (minimum-norm solution initialization)
- $x_{\text{rand}} \sim \mathcal{N}(0, I_{1000})$, (Gaussian iid. randomized zero mean initialization)
- $x_{\text{rand+ridge}} \sim \mathcal{N}(x_{\text{ridge}}, I_{1000})$, (Gaussian iid. randomized with ridge as mean vector initialization)
- $x_{\text{convex}} = \{\ x \mid \underset{x}{\text{argmin}}\ (1/2)\|Gx - b\|_2^2 + \lambda h(Px; \mathcal{K}), q = 1\}$, (convex solution initialization)
- $x_{\text{true}} = \tilde{x}$, (Ground-truth initialization)

We noted that the ground-truth initialization cannot be achieved in practice, we used this as a benchmark to compare with other initialization as the ground-truth initialization should give the best performance. We selected the performance indicators as

1. value of objective function, $(1/2)\|G\hat{x} - b\|_2^2 + \lambda \|\hat{x}\|_{2,1/2}^{(10)}$
2. Relative parameter bias, $\frac{\|\hat{x} - x_{\text{true}}\|_2}{\|x_{\text{true}}\|_2}$.

Table A.1: Converged objective value and relative parameter bias in each initialization over 1000 realizations.

Initialization	Loss	Parameters bias
zero	5.2445	0.5208
ridge	5.2442	0.5208
min-norm	5.2445	0.5208
rand	5.7040	0.7892
ridge+rand	5.7069	0.7931
convex	5.2183	0.4764
Ground truth	**5.2115**	**0.4118**

Results The result is reported in Table A.1. The ground truth initialization gave the best performance on both loss and relative bias, which is expected but cannot be used. The second-best performance is the convex initialization. However, the convex initialization may be too expensive for the large scale setting. The third best is the ridge regression initialization. The complexity is far less than the convex solution and the loss did not drastically different. Therefore, if the least-square solution does not exists, we use ridge regression but if it exists, we can use the least-square solution instead.

Experiment: Non-convex group norm regularization performance in linear regression model

Objective We aim to explore the performance of the non-convex group norm penalty or the $\ell_{p,q}$ group norm penalty against the group lasso or the $\ell_{2,1}$ group-norm penalty. The objective is to find out whether the non-convex regularizer outperformed the group lasso and further compare it with the non-group case, which is lasso and ℓ_q penalty. We expected that the structural prior of both group lasso and $\ell_{p,q}$ group norm penalty to outperform its non-group counterpart.

Setting We generated ground-truth model as,

$$b = G\tilde{x} + \epsilon$$

where $b \in \mathbf{R}^{200}, \tilde{x} \in \mathbf{R}^{1000}$ with SNR of 20dB. We also assumed that $\tilde{x}$ is block-sparse with block size 10 and there are 10 non-zero groups out of 100 groups. We varied the regularization parameter to yield the densest model to the sparsest model. The sparsest model is yielded from the minimum regularization that gives all zero solutions in the group lasso case. The non-convex formulation also varied in the same range because the non-convex formulation tends to be sparser. So that, the regularization bound of the convex case is also a sufficient condition in the non-convex case. The range in ℓ_1, ℓ_q is set in the same sense. We repeated this experiment 10 times. The area under the ROC curve is the indication we selected to measure the performance between these two regression methods.

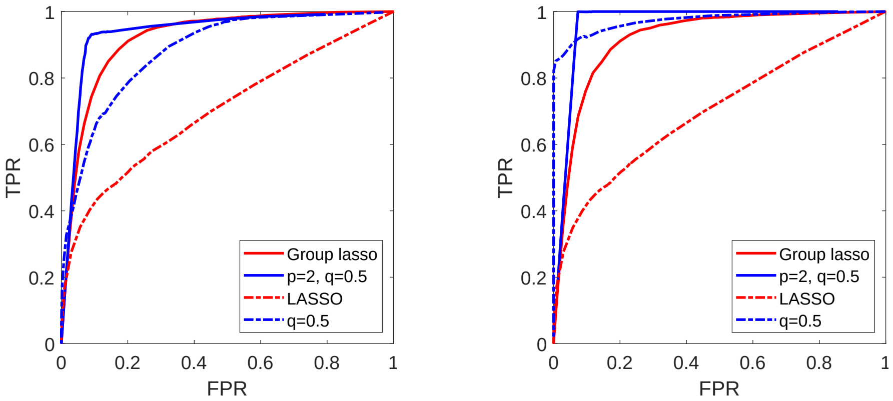


Figure A.1: ROC curve of linear regression using $\ell_{1/2}$, $\ell_{2,1/2}$ regression, lasso and group lasso. nmAPG was initialized by solution of $\ell_{2,1}$ regression (left), ground truth parameters (right).

Results The result in the fig. A.1 suggested that the regularized regression with $\ell_{2,1/2}$ outperformed $\ell_{2,1}$, which agreed with the previous result in the literature. Moreover, both group and non-group non-convex penalties have significant improvement when using the ground-truth model as an initialization as shown in the right plot of Figure A.1. This result indicated that there would be a room for improvement for initial point selection. The effectiveness of group penalties are evidently shown in the left plot of Figure A.1. Even ℓ_q was outperformed by the group lasso. So, if the good initialization cannot be found, the penalty with a grouping structure should be considered first.

Appendix B

MATHEMATICAL DETAIL

B.1 Log-likelihood of multiple VAR models

Consider a problem of finding log-likelihood of models parameters given sets of time-series data $\{y^{(k)}(t)\}_{t=1}^{T}$ for $k = 1, \cdots, K$. The kth VAR model is given by,

$$y^{(k)}(t) = \sum_{r=1}^{p} A_r^{(k)} y^{(k)}(t-r) + \epsilon^{(k)}(t),$$

where $\epsilon^{(k)}(t) \sim \mathcal{N}(0, \Sigma_k)$. For simplicity, we drop the group index k. The log-likelihood for a single measurement at time t is

$$\mathcal{L}(y(t); \Theta) = -\frac{n}{2} \log(2\pi) - \frac{1}{2} \log\det \Sigma - \frac{1}{2} \hat{\epsilon}(t)^T \Sigma^{-1} \hat{\epsilon}(t),$$

where $\Theta = (y(1), \ldots, y(p), A_1, \ldots, A_p, \Sigma)$ and $\hat{\epsilon}(t) = y(t) - \sum_{r=1}^{p} A_r y(t-r)$. We further assume that $\epsilon(t)$ is independent in each t for $t = p+1, \ldots, T$. Therefore, the log-likelihood based on given data points is the sum of log-likelihood in each data point described by

$$\sum_{t=p+1}^{T} \mathcal{L}(y(t); \Theta) = -\frac{nN}{2} \log(2\pi) - \frac{N}{2} \log\det \Sigma - \sum_{t=p+1}^{T} \frac{1}{2} \hat{\epsilon}(t)^T \Sigma^{-1} \hat{\epsilon}(t),$$

where $N = T - p$. We can further rearrange the sum of the quadratic form as

$$\begin{aligned}
\sum_{t=p+1}^{T} \frac{1}{2} \hat{\epsilon}(t)^T \Sigma^{-1} \hat{\epsilon}(t) &= \frac{1}{2} \textbf{tr} \left(\begin{pmatrix} \hat{\epsilon}(p+1)^T \\ \vdots \\ \hat{\epsilon}(T)^T \end{pmatrix} \Sigma^{-1} \begin{pmatrix} \hat{\epsilon}(p+1) & \cdots & \hat{\epsilon}(T) \end{pmatrix} \right) \\
&= \frac{1}{2} \textbf{tr} \left(\Sigma^{-1} \sum_{t=p+1}^{T} \hat{\epsilon}(t) \hat{\epsilon}(t)^T \right) \\
&= \frac{1}{2} \textbf{tr} \left(\Sigma^{-1} (Y - AH)(Y - AH)^T \right)
\end{aligned}$$

where the definition of Y, A, H is given in (2.3), (2.4). When considering K independent models, the log-likelihood is simply the sum of log-likelihood in each model or

$$\sum_{k=1}^{K} \sum_{t=p+1}^{T} \mathcal{L}(y^{(k)}(t); \Theta^{(k)}) = -\frac{nNK}{2} \log(2\pi) - \sum_{k=1}^{K} \frac{N}{2} \log\det \Sigma_k - \sum_{t=p+1}^{T} \frac{1}{2} \hat{\epsilon}^{(k)}(t)^T \Sigma_k^{-1} \hat{\epsilon}^{(k)}(t). \tag{B.1}$$

It is obvious that the maximization problem of (B.1) with respect to all Σ_k is a separable maximization problem in each Σ_k. Therefore, the maximum likelihood estimator of the covariance is

$$\hat{\Sigma}_k = \frac{1}{N}(Y^{(k)} - A^{(k)}H^{(k)})(Y^{(k)} - A^{(k)}H^{(k)})^T, \quad k = 1, \dots, K$$

with the maximum log-likelihood of

$$\underset{\Sigma_1,\dots,\Sigma_K}{\text{maximize}} \sum_{k=1}^{K}\sum_{t=p+1}^{T} \mathcal{L}(y^{(k)}(t); \Theta^{(k)}) = -\frac{nNK}{2}\log(2\pi) - \frac{N}{2}\sum_{k=1}^{K}\log\det(\hat{\Sigma}_k) - \frac{nNK}{2}.$$

B.2 Vectorization of VAR parameters

In Chapter 4, we solved the proposed formulations in the vectorized form as presented in (4.10). Therefore, we provide the detail derivation in the following.

Due to the structure of the penalty functions stated in (4.7)-(4.9), we vectorized K VAR model parameters to be

$$x = (C_{11}, C_{12}, \dots, C_{1n}, C_{21}, \dots, C_{nn})$$

where C_{ij} is presented in (3.5). From this choice of vectorization, the next step is to identify the equivalent sum-square loss term in (3.2) or to find G, b such that

$$\sum_{k=1}^{K} \|Y^{(k)} - A^{(k)}H^{(k)}\|_F^2 \triangleq \|Gx - b\|_2^2, \tag{B.2}$$

where the definition of $Y^{(k)} \in \mathbf{R}^{n\times N}$, $A^{(k)} \in \mathbf{R}^{n\times np}$, $H^{(k)} \in \mathbf{R}^{np\times N}$ is provided in (2.3), (2.4).

To vectorize the sum-square loss function, we derive the case $K = 1$ first and generalizes thereafter. As a single model vectorization, the problem is reduced to the derivation of matrix containing measurements M, v from

$$\|Y - AH\|_F^2 \triangleq \|Mz - v\|_2^2, \tag{B.3}$$

where $M \in \mathbf{R}^{nN\times n^2p}$. We drop the model index, k, to simplify the notation.

We begin by expanding the LHS of (B.3) as

$$\|Y - AH\|_F^2 = \sum_{i=1}^{n}\sum_{t=p+1}^{T}(Y_{it} - \sum_{j=1}^{np} A_{ij}H_{jt})^2. \tag{B.4}$$

We define

$$\mathcal{H}_{jt} = \begin{bmatrix}(H_1)_{jt} & \cdots & (H_p)_{jt}\end{bmatrix} \in \mathbf{R}^{1\times p}, \quad z_{ij} = \begin{bmatrix}(A_1)_{ij} \\ \vdots \\ (A_p)_{ij}\end{bmatrix} \in \mathbf{R}^{p},$$

to rewrite

$$\sum_{j=1}^{np} A_{ij} H_{jt} = \sum_{j=1}^{n} \begin{bmatrix} (H_1)_{jt} & \cdots & (H_p)_{jt} \end{bmatrix} \begin{bmatrix} (A_1)_{ij} \\ \vdots \\ (A_p)_{ij} \end{bmatrix} = \sum_{j=1}^{n} \mathcal{H}_{jt} z_{ij}.$$

By considering the term Y_{it} for $t = 1, \dots, N$, we obtain

$$\begin{bmatrix} \mathcal{H}_{11} & \cdots & \mathcal{H}_{n1} \\ \vdots & \ddots & \vdots \\ \mathcal{H}_{1N} & \cdots & \mathcal{H}_{nN} \end{bmatrix} \begin{bmatrix} z_{i1} \\ \vdots \\ z_{in} \end{bmatrix} - \begin{bmatrix} Y_{i1} \\ \vdots \\ Y_{iN} \end{bmatrix} \triangleq \mathcal{H}\tilde{z}_i - v_i, \tag{B.5}$$

with $\tilde{z}_i \in \mathbf{R}^{np}, v_i \in \mathbf{R}^N$. When varying $i = 1, \dots, n$, we obtained the vectorized loss of k^{th} model as

$$\left\| (I_n \otimes \mathcal{H}) \begin{bmatrix} \tilde{z}_1 \\ \vdots \\ \tilde{z}_n \end{bmatrix} - \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix} \right\|_F^2 \triangleq \|Mz - v\|_2^2, \quad z \in \mathbf{R}^{n^2 p}, v \in \mathbf{R}^{nN}, \tag{B.6}$$

where $(I_n \otimes \mathcal{H}) = \mathbf{diag}(\mathcal{H}, \dots, \mathcal{H})$. At this stage, we see that the fitting term of k^{th} model is,

$$\|M^{(k)} z^{(k)} - v^{(k)}\|_2^2.$$

To cast all K vectorized model into the form in (B.2), we must provide the structure of x, b to construct G. For the vector x, we can also convert $z^{(k)} = (z_{11}^{(k)}, z_{12}^{(k)}, \dots, z_{nn}^{(k)})$ for $k = 1, \dots, K$ to $x = [(z_{11}^{(1)}, \dots, z_{11}^{(K)}), \dots, (z_{nn}^{(1)}, \dots, z_{nn}^{(K)})]$ by the relation,

$$x = \sum_{k=1}^{K} \begin{bmatrix} e_k \otimes z_{11}^{(k)} \\ \vdots \\ e_k \otimes z_{nn}^{(k)} \end{bmatrix}.$$

The vector b can be the row-concatenation of each $v^{(k)}$ so that the resulting b and G is,

$$b = \begin{bmatrix} v^{(1)} \\ \vdots \\ v^{(K)} \end{bmatrix}, G = \begin{bmatrix} G^{(1)} \\ \vdots \\ G^{(K)} \end{bmatrix}.$$

At this point, it can be seen that $G^{(k)}$ is a mapping from every variables in k^{th} VAR model to k^{th} vectorized time-series. For this purpose, we define the matrix $\tilde{\mathcal{H}}^{(k)}$ that initially constructed from $\mathcal{H}^{(k)}$ and a Kronecker product of a standard unit vector e_k in $\mathbf{R}^K$ to ensure that the multiplication between $\tilde{\mathcal{H}}^{(k)}$ and x is the same as $\mathcal{H}^{(k)}$ and $z^{(k)}$. The resulting structure is

$$G^{(k)} = I_n \otimes \tilde{\mathcal{H}}^{(k)} = I_n \otimes \begin{bmatrix} e_k^T \otimes \mathcal{H}_{11}^{(k)} & \cdots & e_k^T \otimes \mathcal{H}_{n1}^{(k)} \\ \vdots & \ddots & \vdots \\ e_k^T \otimes \mathcal{H}_{1N}^{(k)} & \cdots & e_k^T \otimes \mathcal{H}_{nN}^{(k)} \end{bmatrix}. \tag{B.7}$$

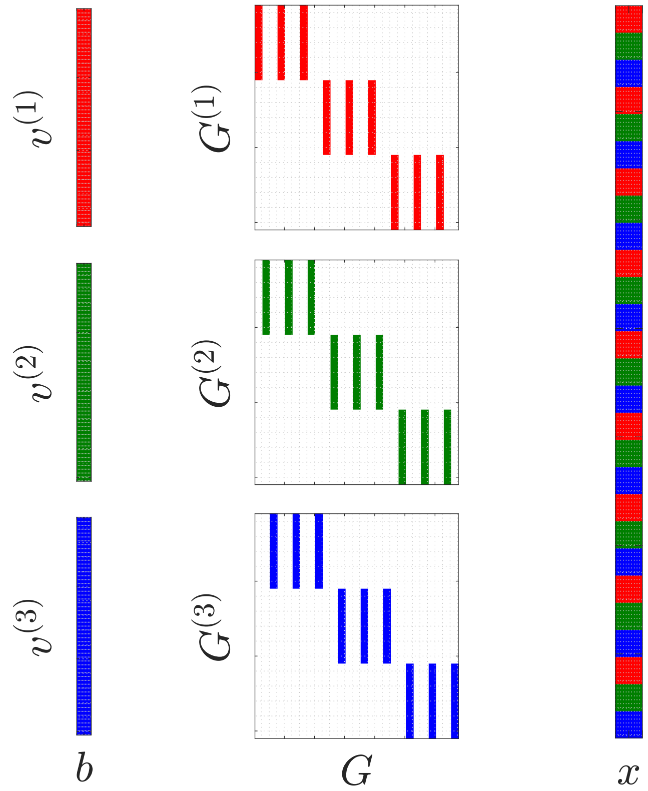


Figure B.1: The structure of G, x, b when $n = 3, p = 5, K = 3$. Each color represents the related elements of k^{th} model: red, green, blue for $k = 1, 2, 3$ respectively.

We yielded the expression of G and b in (B.2). We illustrate the concept of the construction in Figure B.1. It is worth noting that in each $G^{(k)}$, the color stripe was shifted to preserve the relation between $v^{(k)}$ and $G^{(k)}x$.

Hence, we obtain the relation:

$$\sum_{k=1}^{K} \|Y^{(k)} - A^{(k)}H^{(k)}\|_F^2 \triangleq \|Gx - b\|_2^2, \tag{B.8}$$

which is the vectorized sum-square-error with the same definition of x as in (4.2) ■

Appendix C

LIST OF PUBLICATIONS

Most contents of this thesis appeared in three monographs. First, the common GC estimation (without weight penalty), the classification, and the first part of CGN experiment in Chapter 5 were presented in

P. Manomaisaowapak and J. Songsiri, "Learning A Common Granger Causality Network Using A Non-Convex Regularization," *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 1160-1164, doi: 10.1109/ICASSP40776.2020.9054430.

Second, the non-convex CGN formulation and nmAPG algorithm presented in Chapter 4, Chapter 5 were applied to the sub-problem in

P. Manomaisaowapak, A. Nartkulpat and J. Songsiri, Granger causality inference in EEG source connectivity analysis: A state-space approach, 2020 bioRxiv 2020.10.07.329276; doi: 10.1101/2020.10.07.329276 (Accepted for publication in IEEE Transactions on Neural Networks and Learning Systems (TNNLS) journal).

Third, the proposed three formulations and results in both benchmarking experiments and the real data experiments were submitted to ArXiv,

P. Manomaisaowapak and J. Songsiri, Joint estimation of multiple Granger causal networks: Inference of group-level brain connectivity. arXiv:2105.07196 [cs.LG] (Submitted to Neural Networks, under review).

Biography

Parinthorn Manomaisaowapak was born in Bangkok but grown up in Sriracha, Chonburi. He received a Bachelor's degree in Electrical Engineering from Chulalongkorn University in 2018. He pursued a Master's degree in Electrical Engineering with a focus on data analytics at Control System Research Laboratory, Department of Electrical Engineering, Faculty of Engineering, Chulalongkorn University in 2019. His research interests are in the field of causality analysis, sparse signal estimation, non-convex non-smooth optimization.