Model Selection for Multi-Attribute Gaussian Graphical Models

by

Worlasi Kofi Dzam

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
May 2, 2026

Keywords: Model selection, Gaussian graphical models, Stability selection, BIC,
Cross-validation, Inverse covariance estimation, Multi-attribute graphs, Non-convex penalties

Approved by

Jitendra K. Tugnait, Chair, Professor of Electrical and Computer Engineering
Yin Sun, Associate Professor of Electrical and Computer Engineering
Shuai Huang, Assistant Professor of Electrical and Computer Engineering

Abstract

In this thesis, we examine a common challenge in the estimation of conditional independence graphs, namely the selection of the regularization parameter for sparse precision matrix estimation in multi-attribute Gaussian graphical models (MA-GGMs). Even in the classical single-attribute setting, choosing an appropriate model selection criterion remains difficult; the multi-attribute high-dimensional setting further complicates this task. We compare three widely used approaches: stability selection, the Bayesian information criterion, and cross-validation. These techniques are applied to penalized Gaussian likelihood estimation with both convex (lasso) and non-convex (log-sum) penalties. Using extensive simulations on Erdős–Rényi, Barabási–Albert, and chain graphs, we assess how topology, sparsity, sample size, and penalty choice affect the performance of each selector. We further evaluate the methods on a real-world financial time series dataset consisting of multi-attribute stock returns from the S&P 100 index. The empirical results show that stability selection performs well in sparse or small sample regimes; BIC combined with the log-sum penalty provides strong and stable structural recovery; and cross-validation becomes reliable only when the sample size is sufficiently large. These findings provide practical guidance for model selection in high-dimensional multi-attribute graphical models, particularly in applications where accurate recovery of dependence structure is essential.

AI Disclosure Statement

In the preparation of this thesis, the following Artificial Intelligence (AI) tools were used: Claude (Anthropic). This tool was used primarily to review and edit the manuscript for consistency, verify parameter values against source code, and assist with formatting. The author acknowledges full responsibility for the intellectual content of this work and has ensured that all AI-assisted sections have been reviewed and revised for accuracy and appropriate academic style. All AI-generated content was reviewed and validated for relevance, appropriateness, and accuracy before incorporation into the final document to maintain scholarly integrity of this research.

Digital Accessibility Disclosure

In the preparation of this thesis, the following digital accessibility tools were used to ensure this document complies with federal requirements: PAC (PDF Accessibility Checker). The author acknowledges full responsibility for the intellectual content of this work and has made a good faith effort to comply with digital accessibility requirements in publishing, wherein the nature of the content does not significantly change in order to do so. Furthermore, all content has been reviewed and revised to meet these requirements prior to final publication.

Acknowledgments

My sincerest gratitude goes out to my advisor, Professor Jitendra K. Tugnait, for his mentorship and patience throughout the development of this work. I am also grateful to my committee members, Professors Sun and Huang, for their support, suggestions, and encouragement during every stage of my graduate studies. And finally to my family back home in Ghana, who cheer me on continuously. Thank you.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ADMM	Alternating Direction Method of Multipliers
BA	Barabási–Albert (graph model)
BIC	Bayesian Information Criterion
CG	Chain Graph
CI	Conditional Independence
CIG	Conditional Independence Graph
CV	Cross-validation
ER	Erdős–Rényi (graph model)
GGM	Gaussian Graphical Model
LLA	Local Linear Approximation
MA-GGM	Multi-Attribute Gaussian Graphical Model
LSP	Log-Sum Penalty
MLE	Maximum Likelihood Estimation
PD	Positive Definite
SS	Stability Selection
std	Standard Deviation
StARS	Stability Approach to Regularization Selection
Unif	Uniform distribution
F1	F_1-score
HD	Hamming Distance

List of Symbols

p	Number of nodes
m	Number of attributes per node
$d = mp$	Total dimension of the stacked random vector
N	Sample size
$\boldsymbol{z}_i \in \mathbb{R}^m$	Attribute random vector at node i
$\boldsymbol{x}(t) \in \mathbb{R}^{mp}$	Observation at sample index t
$\boldsymbol{\Sigma}$	Covariance matrix of $\boldsymbol{x}$
$\boldsymbol{\Omega}$	Precision matrix ($\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$)
$\boldsymbol{Z}$	ADMM splitting variable in the consensus constraint $\boldsymbol{\Omega} = \boldsymbol{Z}$
$\boldsymbol{\Omega}^{(ij)} \in \mathbb{R}^{m \times m}$	(i, j)-th group of $\boldsymbol{\Omega}$
$\mathcal{G} = (V, \mathcal{E})$	Undirected graph representing conditional independence
$\mathcal{E}$	Edge set of unordered pairs $\{i, j\}$ with $i \neq j$
$V = [p]$	Node set
$\lambda > 0$	Regularization parameter
Λ	Regularization grid
$\alpha \in (0, 1)$	Mixing weight between element-wise and group penalties
ϵ	Log-sum tuning parameter controlling non-convexity
ρ	ADMM penalty parameter
$\|\cdot\|_F$	Frobenius norm
$\mathbf{1}(\cdot)$	Indicator function
$(\widehat{\boldsymbol{\Omega}}_\lambda, \widehat{\boldsymbol{Z}}_\lambda)$	ADMM solution at regularization level λ
$\widehat{\Pi}_{ij}(\lambda)$	Edge selection probability in stability selection
$\widehat{D}(\lambda)$	Instability measure
β	Target instability threshold
p_{thr}	Pruning threshold for stability selection
B	Number of subsamples in stability selection
b_N	Subsample size in stability selection
K	Number of folds in cross-validation
$L_{\text{val}}^{(f)}(\lambda)$	Validation loss on fold f

Chapter 1

Introduction

1.1 Background and Motivation

Graphical models provide a principled framework for representing and learning dependency structure in multivariate data [14]. In an undirected graphical model, conditional dependence relationships are encoded by a graph $\mathcal{G} = (V, \mathcal{E})$, where $V = [p] := \{1, 2, \ldots, p\}$ and $\mathcal{E} \subseteq V \times V$. We exclude self-edges (i, i). Since the graph is undirected, we represent each edge by the unordered pair $\{i, j\}$ and henceforth treat $\mathcal{E}$ as a set of such unordered pairs.

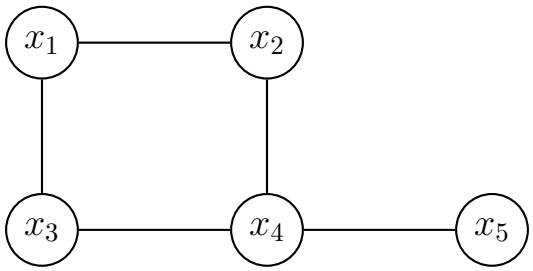


Figure 1.1. Illustration of a graphical model. The nodes (circles) represent random variables $x_1, \ldots, x_5$, and edges (lines) represent conditional dependencies given all other variables. For example, the absence of an edge between x_2 and x_3 indicates that they are conditionally independent given $\{x_1, x_4, x_5\}$.

We denote by $\mathcal{N}_d(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ the d-dimensional Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. Let $\boldsymbol{x} \sim \mathcal{N}_d(\mathbf{0}, \boldsymbol{\Sigma})$. Since $\boldsymbol{\Sigma}$ is a covariance matrix, it is symmetric positive definite, denoted $\boldsymbol{\Sigma} \succ \mathbf{0}$. The corresponding precision matrix is

$$\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}, \tag{1.1}$$

which is also positive definite ($\mathbf{\Omega} \succ \mathbf{0}$), ensuring that the Gaussian density is well defined. A central object of interest is the conditional independence graph (CIG), in which the absence of an edge $\{i, j\} \notin \mathcal{E}$ corresponds to conditional independence between the random variables associated with nodes i and j, given all remaining components of the random vector $\boldsymbol{x}$ (i.e., $\{x_k : k \in V \setminus \{i, j\}\}$) [14]. In a single-attribute GGM, conditional dependence is reflected in the sparsity pattern of the precision matrix [14]. Many modern datasets are multi-attribute, with each node corresponding to a vector of attributes rather than a single scalar variable. The number of attributes per node is denoted by m. Multi-attribute Gaussian graphical models generalize GGMs by associating a random vector with each node and inferring conditional dependencies at the node level [12, 13, 20, 21].

$$\mathbf{\Omega} = \begin{bmatrix} \Omega_{11} & \Omega_{12} & \cdots & \Omega_{1p} \\ \Omega_{21} & \Omega_{22} & \cdots & \Omega_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ \Omega_{p1} & \Omega_{p2} & \cdots & \Omega_{pp} \end{bmatrix}$$

(a) Single-attribute GGM ($m = 1$).

$$\mathbf{\Omega} = \begin{bmatrix} \mathbf{\Omega}^{(11)} & \mathbf{\Omega}^{(12)} & \cdots & \mathbf{\Omega}^{(1p)} \\ \mathbf{\Omega}^{(21)} & \mathbf{\Omega}^{(22)} & \cdots & \mathbf{\Omega}^{(2p)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{\Omega}^{(p1)} & \mathbf{\Omega}^{(p2)} & \cdots & \mathbf{\Omega}^{(pp)} \end{bmatrix}$$

(b) Multi-attribute GGM ($m > 1$).

Figure 1.2. Precision matrix structure in single-attribute and multi-attribute Gaussian graphical models. In the single-attribute case (left), each entry Ω_{ij} is a scalar. In the multi-attribute case (right), each block $\mathbf{\Omega}^{(ij)}$ is an $m \times m$ matrix capturing the dependence structure between the attribute vectors at nodes i and j.

Figure 1.2 highlights the group structure induced by multiple attributes per node. This thesis follows the MA-GGM formulation, notation, and estimation framework in [21].

Under sparsity, a large body of work has proposed methods for learning GGMs, which can be broadly divided into two classes. In Gaussian models, learning the graph structure is equivalent to identifying the nonzero pattern of the precision matrix $\mathbf{\Omega}$. The first class consists of regression-centered or neighborhood selection approaches [11, 17], which estimate the graph structure by fitting a collection of penalized regressions. These methods can be computationally attractive and

scalable, but they do not produce a globally consistent estimate of the inverse covariance matrix and are less naturally suited to estimation and inference via likelihood.

The second class consists of approaches centered around the precision matrix, which estimate the entire inverse covariance matrix by optimizing a Gaussian negative log-likelihood augmented with sparsity-inducing regularization [1, 9, 20, 21]. Although these methods involve higher-dimensional optimization problems, they tend to yield a coherent global model, naturally extend to the group-structured and multi-attribute settings, and facilitate principled model selection using likelihood as a criterion. This thesis adopts this latter approach. Recent studies show that different model selection criteria can behave differently depending on whether the goal is predictive accuracy, meaning good performance on unseen data, or structural recovery, meaning correct identification of the underlying graph structure [6]. This thesis investigates this distinction in the multi-attribute setting.

1.2 Multi-Attribute Gaussian Graphical Model

Let each node $j \in V$ be associated with an m-dimensional, zero-mean Gaussian random vector $\boldsymbol{z}_j \in \mathbb{R}^m$. Define the stacked random vector as

$$\boldsymbol{x} = [\boldsymbol{z}_1^\top \ \boldsymbol{z}_2^\top \ \cdots \ \boldsymbol{z}_p^\top]^\top \in \mathbb{R}^{mp}, \tag{1.2}$$

and then assume that

$$\boldsymbol{x} \sim \mathcal{N}_{mp}(\mathbf{0}, \boldsymbol{\Sigma}), \qquad \boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1} \succ 0. \tag{1.3}$$

Following [21], the stacked representation yields an equivalent scalar-level formulation of the model. The precise construction of this representation, its relationship to node-level conditional independence, and the formal definitions are developed in Chapter 2. The corresponding precision matrix admits a natural group structure, with groups corresponding to pairs of nodes. In MA-GGMs, conditional dependencies are encoded at the node-level through the off-diagonal groups

of the precision matrix. Under sparsity, entire groups may "vanish" when no direct node-level dependence is present, leading to a group-sparse structure [20, 21].

1.3 Problem Statement

We observe N i.i.d. zero-mean samples $\{\boldsymbol{x}(t)\}_{t=1}^{N}$ and form the sample covariance matrix

$$\widehat{\boldsymbol{\Sigma}} = \frac{1}{N}\sum_{t=1}^{N} \boldsymbol{x}(t)\boldsymbol{x}^{\top}(t). \tag{1.4}$$

The objective of this thesis is to recover the underlying conditional independence structure among nodes from the observed data. To achieve this, we estimate the precision matrix $\boldsymbol{\Omega}$ under sparsity-inducing penalties and compare how different model selection criteria affect the quality of the recovered graph structure.

1.4 Notation

For a finite set V, $|V|$ denotes its cardinality. For a matrix $\boldsymbol{A} \in \mathbb{R}^{p\times p}$, $\lambda_{\min}(\boldsymbol{A})$ and $\lambda_{\max}(\boldsymbol{A})$ denote its minimum and maximum eigenvalues, respectively. The determinant and trace of $\boldsymbol{A}$ are denoted by $\det(\boldsymbol{A})$ and $\mathrm{tr}(\boldsymbol{A})$. The Frobenius norm is $\|\boldsymbol{A}\|_F = \sqrt{\mathrm{tr}(\boldsymbol{A}^{\top}\boldsymbol{A})}$. The sign function $\mathrm{sign}(x)$ returns $+1$ if $x > 0$, -1 if $x < 0$, and 0 if $x = 0$. The entry A_{ij} (also denoted $[\boldsymbol{A}]_{ij}$) is the (i, j)th entry of $\boldsymbol{A}$. For a group-partitioned matrix, $\boldsymbol{A} = [\boldsymbol{A}_{ij}] \in \mathbb{R}^{mp\times mp}$ with $\boldsymbol{A}_{ij} \in \mathbb{R}^{m\times m}$. The number of observations is denoted by N, and the sample covariance matrix is defined in (1.4).

1.5 Organization of the Thesis

The remainder of this thesis is organized as follows. Chapter 2 establishes the statistical foundations of Gaussian graphical models, including the relationship between conditional independence and the structure of the precision matrix in single-attribute and multi-attribute settings. Chapter 3 presents the sparse-group penalized likelihood framework used to estimate MA-GGM precision

matrices, along with the penalty choices and optimization approach. Chapter 4 introduces the model selection criteria studied in this work, namely, stability selection, the Bayesian information criterion, and cross-validation. Chapter 5 describes the simulation design used for the comparative study. Chapter 6 reports the numerical results of the synthetic experiments, and Chapter 7 applies the methodology to a real-world financial time series dataset from the S&P 100 index. Finally, Chapter 8 summarizes the main findings, discusses practical implications, and outlines possible directions for future work. Appendix A provides the detailed algorithmic framework and implementation details for the ADMM solver, LLA scheme, and each model selection criterion, while Appendix B documents the computational environment, parameter settings, and the MATLAB scripts used to produce all results.

Chapter 2

Statistical Model and Conditional Independence Structure

This chapter provides the probabilistic foundations underlying Gaussian graphical models. In particular, it establishes the relationship between conditional independence and the sparsity structure of the precision matrix in single-attribute and multi-attribute settings.

2.1 Conditional Independence and the Pairwise Markov Property

Recall the undirected graph $\mathcal{G} = (V, \mathcal{E})$ introduced in Chapter 1, where each node $i \in V$ is associated with an m-dimensional random vector $\boldsymbol{z}_i \in \mathbb{R}^m$. We interpret $\mathcal{G}$ as a conditional independence graph (CIG): $\{i, j\} \notin \mathcal{E}$ if and only if $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ are conditionally independent given the remaining nodes. For distinct nodes $i, j \in V$, conditional independence is defined by the conditional density factorization:

$$f(\boldsymbol{z}_i, \boldsymbol{z}_j \mid \{\boldsymbol{z}_k : k \in V \setminus \{i, j\}\}) = f(\boldsymbol{z}_i \mid \{\boldsymbol{z}_k : k \in V \setminus \{i, j\}\}) f(\boldsymbol{z}_j \mid \{\boldsymbol{z}_k : k \in V \setminus \{i, j\}\}). \tag{2.1}$$

An undirected graph $\mathcal{G}$ is said to represent the conditional independence structure of $\{\boldsymbol{z}_i\}_{i \in V}$ if $\{i, j\} \notin \mathcal{E}$ if and only if the conditional density factorization in (2.1) holds. This is the pairwise Markov property [14]. For distributions with strictly positive density, including Gaussian distributions, the pairwise, local, and global Markov properties are equivalent [14].

2.2 Gaussian Graphical Models and Precision Matrices

Consider a single-attribute Gaussian graphical model, with random vector $\boldsymbol{x} = [x_1, \ldots, x_p]^\top \sim \mathcal{N}_p(\mathbf{0}, \boldsymbol{\Sigma})$ and precision matrix $\boldsymbol{\Omega} = \boldsymbol{\Sigma}^{-1}$. A key property of multivariate Gaussian distributions is that conditional independence relationships are completely determined by the precision matrix. The following classical result establishes the relationship between conditional independence and the sparsity pattern of $\boldsymbol{\Omega}$ in Gaussian models. Here, the notation $x_i \perp\!\!\!\perp x_j \mid \{x_k : k \in V \setminus \{i, j\}\}$ indicates that the scalar variables x_i and x_j are conditionally independent given the remaining variables.

Proposition 2.1 (Gaussian conditional independence [14])**.** *For distinct* $i, j \in V$

$$\Omega_{ij} = 0 \quad \Longleftrightarrow \quad x_i \perp\!\!\!\perp x_j \mid \{x_k : k \in V \setminus \{i, j\}\}. \tag{2.2}$$

This is because, in the Gaussian case, the conditional density $f(x_i, x_j \mid \{x_k : k \in V \setminus \{i, j\}\})$ is itself Gaussian. Its precision matrix is obtained via the Schur complement of $\boldsymbol{\Sigma}$ with respect to the indices $\{i, j\}$. The off-diagonal entry of this conditional precision matrix is determined by Ω_{ij}. Hence, $\Omega_{ij} = 0$ if and only if the conditional density factorizes, thereby establishing conditional independence. Therefore, learning the conditional independence graph in a Gaussian graphical model is equivalent to identifying the zero pattern of $\boldsymbol{\Omega}$.

2.3 Multi-Attribute Gaussian Graphical Models

In a multi-attribute Gaussian graphical model, each node $i \in V$ is associated with an m-dimensional Gaussian random vector $\boldsymbol{z}_i \in \mathbb{R}^m$. Define the stacked vector $\boldsymbol{x} = [\boldsymbol{z}_1^\top \ \cdots \ \boldsymbol{z}_p^\top]^\top \in \mathbb{R}^{mp}$. To analyze conditional independence at the node level, it is useful to associate the stacked vector $\boldsymbol{x}$ with an enlarged single-attribute graph $\bar{\mathcal{G}} = (\bar{V}, \bar{\mathcal{E}})$, where $\bar{V} = [mp]$ indexes the scalar components of $\boldsymbol{x}$ and $\bar{\mathcal{E}} \subseteq \bar{V} \times \bar{V}$ denotes the corresponding edge set. Each scalar component of $\boldsymbol{x}$ is treated as a node in $\bar{\mathcal{G}}$. Since $\bar{\mathcal{G}}$ is a single-attribute GGM, Proposition 2.1 applies to its scalar entries:

$[\boldsymbol{\Omega}]_{rs} = 0$ if and only if the rth and sth scalar components of $\boldsymbol{x}$ are conditionally independent given all others. Node-level conditional independence between nodes i and j then holds if and only if *all* scalar entries in the corresponding group $\boldsymbol{\Omega}^{(ij)}$ are zero, which motivates examining the group structure of $\boldsymbol{\Omega}$. We partition $\boldsymbol{\Omega} \in \mathbb{R}^{mp \times mp}$ into $p \times p$ groups of size $m \times m$, and denote the (i, j)th group by $\boldsymbol{\Omega}^{(ij)}$, defined entrywise as

$$[\boldsymbol{\Omega}^{(ij)}]_{st} = [\boldsymbol{\Omega}]_{(i-1)m+s,\,(j-1)m+t}, \qquad s, t \in \{1, \ldots, m\}. \tag{2.3}$$

That is, the group $\boldsymbol{\Omega}^{(ij)}$ is obtained by extracting the $m \times m$ submatrix of $\boldsymbol{\Omega}$ corresponding to the m attributes of node i (rows $(i-1)m+1$ to im) and the m attributes of node j (columns $(j-1)m+1$ to jm). Consequently, for distinct $i, j \in V$, node-level conditional independence is characterized by the structure of the group $\boldsymbol{\Omega}^{(ij)}$. Here, $\boldsymbol{z}_i \perp\!\!\!\perp \boldsymbol{z}_j \mid \{\boldsymbol{z}_k : k \in V \setminus \{i, j\}\}$ denotes that the random vectors $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ are conditionally independent given the remaining nodes. In particular, the following characterization holds [21]:

$$\boldsymbol{z}_i \perp\!\!\!\perp \boldsymbol{z}_j \mid \{\boldsymbol{z}_k : k \in V \setminus \{i, j\}\} \quad \Longleftrightarrow \quad \|\boldsymbol{\Omega}^{(ij)}\|_F = 0. \tag{2.4}$$

Accordingly, the multi-attribute conditional independence graph $\mathcal{G} = (V, \mathcal{E})$ satisfies:

$$\{i, j\} \notin \mathcal{E} \quad \Longleftrightarrow \quad \|\boldsymbol{\Omega}^{(ij)}\|_F = 0, \qquad i \neq j. \tag{2.5}$$

This result extends Proposition 2.1 to the multi-attribute setting.

2.4 Summary

This chapter established the probabilistic foundations of Gaussian graphical models and the relationship between conditional independence and the precision matrix. In single-attribute models, conditional independence corresponds to zeros in individual precision matrix entries. In multi-attribute models, node-level conditional independence is encoded by vanishing Frobenius norms of off-diagonal precision matrix groups. This group-sparse structure motivates the penalized likelihood framework developed in the next chapter.

Chapter 3

Penalized Likelihood Estimation for MA-GGMs

In this chapter, we present the penalized Gaussian likelihood framework used to estimate the precision matrix in multi-attribute Gaussian graphical models. This formulation follows the work of [21] and provides the estimation engine for the model selection criteria in Chapter 4.

3.1 Gaussian Negative Log-Likelihood

Let $\{\boldsymbol{x}(t)\}_{t=1}^{N}$ denote i.i.d. observations from the MA-GGM defined in Chapter 2, with the sample covariance matrix defined in (1.4). Parameterizing the Gaussian log-likelihood in terms of $\boldsymbol{\Omega}$, the negative log-likelihood, up to an additive constant [21], is

$$L(\boldsymbol{\Omega}) = -\ln\det(\boldsymbol{\Omega}) + \operatorname{tr}\left(\widehat{\boldsymbol{\Sigma}}\,\boldsymbol{\Omega}\right), \qquad \boldsymbol{\Omega} \succ 0. \tag{3.1}$$

In high-dimensional settings, $\widehat{\boldsymbol{\Sigma}}$ may be ill-conditioned or singular, making unpenalized maximum likelihood estimation unstable or infeasible [1, 9]. This motivates penalized likelihood methods that impose sparsity on $\boldsymbol{\Omega}$, making the problem well-posed and solvable in high dimensions.

3.2 Sparse-Group Regularization for MA-GGMs

As established in Chapter 2, node-level conditional independence in MA-GGMs is encoded by the group structure of the precision matrix; thus, $\{i, j\} \notin \mathcal{E}$ corresponds to a vanishing off-diagonal

group $\mathbf{\Omega}^{(ij)}$. Accordingly, the needed sparsity should be promoted both at the level of individual entries and at the level of entire groups. To this end, we consider a sparse-group penalized version of the negative log-likelihood. Let $\rho_\lambda(\cdot)$ represent a scalar penalty function which promotes sparsity, parameterized by $\lambda > 0$ and applied to $|u|$ for $u \in \mathbb{R}$, following [21]. Two choices are considered in this thesis: the convex lasso penalty and the non-convex log-sum penalty. The convex lasso penalty is defined by [21] as

$$\rho_\lambda(u) = \lambda|u|, \qquad \lambda > 0. \tag{3.2}$$

As a representative non-convex alternative, the log-sum penalty (LSP), also given by [21], is

$$\rho_\lambda(u) = \lambda\epsilon \ln\left(1 + \frac{|u|}{\epsilon}\right), \qquad \lambda > 0, \;\; 1 \gg \epsilon > 0. \tag{3.3}$$

The parameter ϵ controls shrinkage bias: smaller ϵ reduces bias on larger coefficients while increasing non-convexity, whereas larger ϵ yields behavior closer to the convex lasso penalty. Using $\rho_\lambda(\cdot)$, the element-wise penalty is defined over the enlarged index set $[mp]$ as

$$P_e(\mathbf{\Omega}) = \sum_{\substack{r,s\in[mp]\\ r\neq s}} \rho_\lambda\Big(\big|[\mathbf{\Omega}]_{rs}\big|\Big). \tag{3.4}$$

Here, $(r, s) \in [mp] \times [mp]$ denote scalar indices of the expanded precision matrix. The element-wise penalty $P_e(\mathbf{\Omega})$ penalizes all off-diagonal scalar entries of $\mathbf{\Omega} \in \mathbb{R}^{mp\times mp}$ while leaving the diagonal unpenalized, thereby encouraging individual entries to be exactly zero. Next, we partition $\mathbf{\Omega} \in \mathbb{R}^{mp\times mp}$ into $p \times p$ groups $\mathbf{\Omega}^{(ij)} \in \mathbb{R}^{m\times m}$. The group penalty is then defined as

$$P_g(\mathbf{\Omega}) = m \sum_{\substack{i,j\in[p]\\ i\neq j}} \rho_\lambda\Big(\big\|\mathbf{\Omega}^{(ij)}\big\|_F\Big), \tag{3.5}$$

where $\|\cdot\|_F$ denotes the Frobenius norm and the scaling factor m follows [21]. This penalty pushes entire off-diagonal groups to shrink to zero and therefore directly promotes sparsity of the node-level edge set. Combining the likelihood and penalty terms, we define the sparse-group penalized negative log-likelihood as

$$\bar{L}(\mathbf{\Omega}) = L(\mathbf{\Omega}) + \alpha P_e(\mathbf{\Omega}) + (1-\alpha) P_g(\mathbf{\Omega}), \tag{3.6}$$

where $\alpha \in [0,1]$ controls the balance between element-wise and group-wise penalties. The resulting estimator is obtained by minimizing $\bar{L}$:

$$\widehat{\mathbf{\Omega}} = \arg\min_{\mathbf{\Omega} \succ 0} \ \bar{L}(\mathbf{\Omega}). \tag{3.7}$$

This formulation corresponds to the sparse-group penalized MA-GGM estimator in [21].

Optimization remark: When $\rho_\lambda(\cdot)$ is the lasso penalty (3.2), $\bar{L}(\mathbf{\Omega})$ is convex. When $\rho_\lambda(\cdot)$ is the log-sum penalty (3.3), the objective becomes non-convex. Following [21], the non-convex problem is addressed using a local linear approximation (LLA) scheme, which generates a sequence of convex weighted sparse-group subproblems, each solved using the alternating direction method of multipliers (ADMM). ADMM introduces an auxiliary splitting variable $\boldsymbol{Z} \in \mathbb{R}^{mp \times mp}$ and enforces the consensus constraint $\mathbf{\Omega} = \boldsymbol{Z}$, allowing the smooth likelihood and nonsmooth penalty terms to be optimized in alternating steps. Specifically, $\mathbf{\Omega}$ is updated in the likelihood step and remains positive definite, while $\boldsymbol{Z}$ is updated in the penalty step via element-wise thresholding and groupwise shrinkage. Since the penalty step creates exact zeros, sparsity patterns, edge sets, and related selector quantities are computed from $\boldsymbol{Z}$ throughout this thesis. At convergence, the consensus constraint enforces $\mathbf{\Omega} = \boldsymbol{Z}$, so this is consistent with the final precision estimate. We denote the converged ADMM iterates at regularization level λ by $(\widehat{\mathbf{\Omega}}_\lambda, \widehat{\boldsymbol{Z}}_\lambda)$. The necessary algorithmic details, including the ADMM update steps and the LLA reweighting scheme, are presented in Appendix A.

3.3 Summary

In this chapter, we presented the penalized likelihood framework used to estimate the MA-GGM precision matrix. A sparse-group penalized negative log-likelihood was defined by combining the Gaussian negative log-likelihood with both an element-wise penalty and a group-level penalty. Both lasso and LSP were considered. For the lasso penalty, optimization is carried out via ADMM; and for LSP, ADMM is combined with the LLA scheme. With the estimation framework in place, the next chapter introduces the model selection criteria used to choose the regularization parameter λ.

Chapter 4

Model Selection Criteria for MA-GGMs

In this chapter, we formally introduce the model selection criteria used to choose the regularization parameter λ for MA-GGMs estimated via the penalized likelihood framework in Chapter 3. We study three approaches: stability selection (SS), the Bayesian information criterion (BIC), and cross-validation (CV). For each, we present its implementation, theoretical foundations, and the assumptions under which it is appropriate for the estimation problem considered in this thesis.

4.1 Regularization Grid Construction

For each dataset, regularization parameters are selected over finite grids. For SS and CV, we construct a grid $\Lambda = \{\lambda_1, \ldots, \lambda_T\}$ with $0 < \lambda_1 < \cdots < \lambda_T < \infty$ and $T = 8$ grid points, logarithmically spaced. An upper endpoint λ_T is determined via a bisection procedure as the smallest regularization value for which the estimated graph is nearly empty, in the sense that its number of selected edges falls below a preset sparsity threshold used in the bisection routine. This choice ensures that the grid includes a clearly sparse regime and captures the transition from sparse to moderately dense graphs. We then set $\lambda_1 = \lambda_T/5$ and take the remaining values by logarithmic spacing between λ_1 and λ_T.

For BIC, we use a separate grid with the same size ($T = 8$) but a different range. Starting from the same bisection-derived endpoint λ_T, we set $\lambda_T^{\mathrm{BIC}} = \lambda_T/2$ and $\lambda_1^{\mathrm{BIC}} = \lambda_T^{\mathrm{BIC}}/10$, and then define $\Lambda_{\mathrm{BIC}} = \{\lambda_1^{\mathrm{BIC}} < \cdots < \lambda_T^{\mathrm{BIC}}\}$ by logarithmic spacing. This BIC grid concentrates the search on

moderately sparse regimes and avoids overly dense solutions that are typically penalized heavily by BIC.

Logarithmic spacing is used because the changes in the estimated edge set tend to occur over multiplicative, rather than additive, changes in λ. In particular, the sparsity pattern can change substantially over a narrow range of small λ values, while becoming less sensitive for larger λ. A log grid therefore covers a wide dynamic range with few points and provides useful resolution in sparsity regimes where selection is the most sensitive.

4.2 Stability Selection

SS is a model selection framework designed to identify graph structures that are reproducible under data perturbations, rather than those that merely optimize a single sample likelihood criterion [15, 18]. The underlying idea is intuitive: if an edge appears consistently across many random subsets of the data, it is likely a true feature of the underlying graph rather than an artifact of the particular sample. By measuring this consistency across subsamples, SS identifies edges that are stable and therefore more likely to reflect genuine conditional dependencies. We follow the StARS framework of [15] for regularization parameter selection and supplement it with a pruning step based on selection probabilities as in [18] to further control false positives.

First, consider a given regularization grid Λ constructed as described in Section 4.1, and let B denote the number of subsamples. For each subsample $b = 1, \ldots, B$, a subset of observations of size $b_N = \lfloor 10\sqrt{N} \rfloor$ is drawn uniformly without replacement from the original N observations $\{\boldsymbol{x}(t)\}_{t=1}^{N}$, following [15]. This choice balances two requirements: the subsample must be large enough to yield reliable estimates ($b_N \to \infty$), yet small enough relative to N to detect sampling variability ($b_N/N \to 0$), as required by [15]. From each subsample, a sample covariance matrix is computed and the penalized likelihood problem is solved, producing an estimated precision matrix $\widehat{\boldsymbol{\Omega}}_{\lambda}^{(b)}$ and its associated estimated graph. As noted in Chapter 3, edge sets are determined from the ADMM splitting variable $\widehat{\boldsymbol{Z}}_{\lambda}^{(b)}$ using groupwise Frobenius norms of the off-diagonal groups.

To record whether a given edge appears in a particular subsample estimate, we introduce a binary edge selection indicator $\widehat{A}$. For each node pair $\{i, j\}$ with $i < j$, let $\widehat{\boldsymbol{Z}}_{\lambda}^{(b)(ij)}$ denote the (i, j)th $m \times m$ group of the ADMM splitting variable from subsample b. We define

$$\widehat{A}_{ij}^{(b)}(\lambda) = \mathbf{1}\Big\{ \|\widehat{\boldsymbol{Z}}_{\lambda}^{(b)(ij)}\|_F > 0 \Big\}, \tag{4.1}$$

where $\widehat{A}_{ij}^{(b)}(\lambda) = 1$ indicates that an edge between nodes i and j is selected in subsample b, and $\widehat{A}_{ij}^{(b)}(\lambda) = 0$ otherwise. Aggregating these binary decisions across subsamples yields an empirical measure of how frequently each edge is selected. For each edge $\{i, j\}$, we count the number of subsamples in which it was selected, then divide by the total number of subsamples B to obtain its selection probability $\widehat{\Pi}$. The selection probability of a given edge at regularization level λ is defined as

$$\widehat{\Pi}_{ij}(\lambda) = \frac{1}{B} \sum_{b=1}^{B} \widehat{A}_{ij}^{(b)}(\lambda), \qquad i < j. \tag{4.2}$$

Values of $\widehat{\Pi}_{ij}(\lambda)$ close to one indicate edges that are consistently recovered across subsamples, while values close to zero indicate unstable edges. To summarize variability at a given λ, StARS uses the instability measure $\widehat{D}(\lambda)$, which quantifies the average disagreement across subsamples. Since there are $p(p-1)/2$ possible edges among p nodes, the normalization ensures that $\widehat{D}(\lambda)$ is an average instability measure. Following [15], the measure is defined as

$$\widehat{D}(\lambda) = \frac{2}{p(p-1)} \sum_{i<j} 2\widehat{\Pi}_{ij}(\lambda)\big(1 - \widehat{\Pi}_{ij}(\lambda)\big), \tag{4.3}$$

which satisfies $\widehat{D}(\lambda) \in [0, 0.5]$ as in [15]. For very large λ, no edges are selected and $\widehat{D}(\lambda)$ is close to zero; for very small λ, almost all edges are selected and $\widehat{D}(\lambda)$ again approaches zero. In finite samples, $\widehat{D}(\lambda)$ may therefore vary non-monotonically along the regularization grid [15]. Since the objective is to identify stable sparse graphs, it is desirable to enforce monotonicity, that is, to ensure that instability does not artificially decrease as the graph becomes denser, which could lead to selecting an overly dense model. This is achieved via a monotone correction applied along the

regularization grid, defined as:

$$\widehat{D}_{\mathrm{mono}}(\lambda_t) = \max_{s \geq t} \widehat{D}(\lambda_s), \qquad t = 1, \ldots, T, \tag{4.4}$$

so that instability does not decrease as λ decreases. Given a target instability threshold $\beta \in (0, 1)$, the stability selection estimate chooses the smallest regularization level whose monotonized instability does not exceed β:

$$\lambda_{\mathrm{SS}} = \min\big\{\lambda_t \in \Lambda : \ \widehat{D}_{\mathrm{mono}}(\lambda_t) \leq \beta\big\}. \tag{4.5}$$

After λ_{SS} is chosen, we form two graphs that use different information sources. The first is the unpruned SS graph, obtained by solving (3.6) using all N observations at λ_{SS}, yielding a final estimate $\widehat{\boldsymbol{Z}}_{\lambda_{\mathrm{SS}}}$. Each edge $\{i, j\}$ in this graph is assigned a continuous weight $\|\widehat{\boldsymbol{Z}}^{(ij)}_{\lambda_{\mathrm{SS}}}\|_F$, the Frobenius norm of the corresponding precision matrix block, which reflects the estimated strength of conditional dependence.

The second is the pruned SS graph, which is not obtained by trimming $\widehat{\boldsymbol{A}}_{\mathrm{unpruned}}$. Instead, it is constructed directly from stability weights computed across subsamples at λ_{SS}. Specifically, we compute the selection probabilities $\widehat{\Pi}_{ij}(\lambda_{\mathrm{SS}})$ as in (4.2) and retain an edge only if it meets the pruning threshold:

$$\widehat{\mathcal{E}}_{\mathrm{pruned}} = \Big\{\{i, j\} : \widehat{\Pi}_{ij}(\lambda_{\mathrm{SS}}) \geq p_{\mathrm{thr}}\Big\}. \tag{4.6}$$

This yields a binary adjacency matrix $\widehat{\boldsymbol{A}}_{\mathrm{pruned}}$ where

$$[\widehat{\boldsymbol{A}}_{\mathrm{pruned}}]_{ij} = \begin{cases} 1, & \widehat{\Pi}_{ij}(\lambda_{\mathrm{SS}}) \geq p_{\mathrm{thr}}, \\ 0, & \text{otherwise}, \end{cases} \tag{4.7}$$

for $i < j$ (and symmetrized for $i > j$). Thus, the pruned graph encodes whether an edge is reproducibly detected per the threshold. For edges retained in the pruned graph, the associated

strength of conditional dependence is taken from the full-sample ADMM solution, that is, the groupwise Frobenius norm $\|\widehat{\boldsymbol{Z}}^{(ij)}_{\lambda_{\mathrm{SS}}}\|_F$ used in the unpruned graph. The pruned version serves as a more robust estimator, helping to control false positives by retaining only those edges that appear consistently across subsamples. Additional implementation details are provided in Appendix A.3.

4.3 Bayesian Information Criterion

BIC selects the regularization parameter λ by balancing goodness of fit with model complexity. For each $\lambda \in \Lambda_{\mathrm{BIC}}$, let $(\widehat{\boldsymbol{\Omega}}_\lambda, \widehat{\boldsymbol{Z}}_\lambda)$ denote the ADMM solution obtained from (3.6). As in the SS case, edge sets and sparsity-based quantities are determined from $\widehat{\boldsymbol{Z}}_\lambda$ (see Chapter 3).
To quantify complexity, we adopt the enlarged scalar representation $\widehat{\boldsymbol{Z}}_\lambda \in \mathbb{R}^{mp\times mp}$ and define the estimated enlarged edge set

$$\widehat{\widetilde{\mathcal{E}}}(\lambda) = \big\{(r,s) \in [mp] \times [mp] : r \neq s,\ |[\widehat{\boldsymbol{Z}}_\lambda]_{rs}| > 0\big\}. \tag{4.8}$$

Since $\widehat{\boldsymbol{Z}}_\lambda$ is symmetric, each nonzero off-diagonal entry contributes one free parameter, and the number of free off-diagonal parameters is $\frac{1}{2}|\widehat{\widetilde{\mathcal{E}}}(\lambda)|$. Diagonal entries are unpenalized and contribute an additive constant across λ; they are omitted from the comparison. The resulting BIC score is

$$\mathrm{BIC}(\lambda) = \mathrm{tr}\Big(\widehat{\boldsymbol{\Sigma}}\,\widehat{\boldsymbol{Z}}_\lambda\Big) - \ln\det\Big(\widehat{\boldsymbol{Z}}_\lambda\Big) + \frac{\ln N}{N}\,\frac{|\widehat{\widetilde{\mathcal{E}}}(\lambda)|}{2}. \tag{4.9}$$

This formulation penalizes dense graphs by increasing the cost of each additional nonzero off-diagonal entry in the enlarged precision matrix, thereby favoring sparser graphical models. At convergence, the ADMM consensus constraint enforces $\boldsymbol{\Omega} = \boldsymbol{Z}$, so $\widehat{\boldsymbol{Z}}_\lambda$ is positive definite and the log-determinant in (4.9) is well-defined. The BIC-selected regularization level is $\lambda_{\mathrm{BIC}} = \arg\min_{\lambda\in\Lambda_{\mathrm{BIC}}} \mathrm{BIC}(\lambda)$. Once λ_{BIC} is selected, the penalized likelihood problem is solved on the

full dataset at this regularization level. Edges are declared present using the same groupwise Frobenius norm rule described for SS, with edge weights given by the corresponding group norms. Extensions such as EBIC introduce stronger penalties for ultra-high dimensional model spaces, but are not considered here [5, 8]. Additional details are provided in Appendix A.4.

4.4 Cross-Validation

CV is a widely used approach that selects λ by optimizing predictive performance on held-out data [19]. The underlying idea is to repeatedly partition the data into training and validation subsets: the model is fitted on the training portion and evaluated on the held-out portion. The regularization parameter that yields the best average predictive performance across all partitions is selected. Unlike SS and BIC, CV targets predictive fit rather than structural recovery, meaning that it seeks the λ that best predicts unseen data rather than the λ that most accurately identifies the true edge set. We employ K-fold cross-validation, where $\{1, \dots, N\}$ is partitioned into K disjoint folds $\mathcal{I}_1, \dots, \mathcal{I}_K$. For each $\lambda \in \Lambda$ and each fold f, the training index set is $\mathcal{I}_{\text{tr}}^{(f)} = \{1, \dots, N\} \setminus \mathcal{I}_f$ and the validation index set is $\mathcal{I}_{\text{val}}^{(f)} = \mathcal{I}_f$. Using the training data, we compute $\widehat{\boldsymbol{\Sigma}}_{\text{tr}}^{(f)}$ and obtain $\widehat{\boldsymbol{\Omega}}_{\lambda}^{(f)}$ by solving (3.6). Using the validation fold, we compute $\widehat{\boldsymbol{\Sigma}}_{\text{val}}^{(f)}$ and define the validation loss as

$$L_{\text{val}}^{(f)}(\lambda) = \operatorname{tr}\left(\widehat{\boldsymbol{\Sigma}}_{\text{val}}^{(f)} \, \widehat{\boldsymbol{\Omega}}_{\lambda}^{(f)}\right) - \ln\det\left(\widehat{\boldsymbol{\Omega}}_{\lambda}^{(f)}\right). \tag{4.10}$$

The CV score is obtained by averaging across folds:

$$\mathrm{CV}(\lambda) = \frac{1}{K} \sum_{f=1}^{K} L_{\text{val}}^{(f)}(\lambda), \qquad \lambda_{\mathrm{CV}} = \arg\min_{\lambda \in \Lambda} \mathrm{CV}(\lambda). \tag{4.11}$$

After selecting λ_{CV}, the model is fit on all N observations at this regularization level, and the final graph is obtained using the same edge detection and weighting procedure as for SS and BIC. Thus, while validation uses $\boldsymbol{\Omega}$ to assess predictive performance, the final structural estimate relies on $\boldsymbol{Z}$ for edge detection.

Remark on the validation loss: The log-determinant in (4.10) is evaluated using $\widehat{\boldsymbol{\Omega}}_{\lambda}^{(f)}$ rather than $\widehat{\boldsymbol{Z}}_{\lambda}^{(f)}$. Although the ADMM consensus constraint enforces $\boldsymbol{\Omega} = \boldsymbol{Z}$ at convergence, the $\boldsymbol{\Omega}$ update directly enforces positive definiteness at every iteration through its closed-form eigenvalue solution (see Appendix A), making it the natural choice for evaluating the log-determinant in the validation loss.

Because CV evaluates predictive fit rather than graph recovery, minimizing $\mathrm{CV}(\lambda)$ does not necessarily correspond to recovering the true conditional independence structure. This distinction is particularly important in high-dimensional MA-GGMs and is reflected in the empirical comparisons in Chapters 6 and 7. Additional implementation details are provided in Appendix A.5.

4.5 Summary

SS emphasizes reproducibility under subsampling and supports pruning via selection probabilities; BIC balances likelihood fit and complexity via parameter counting; and CV selects λ by minimizing held-out negative log-likelihood. Each criterion reflects a different notion of model quality: SS targets edge stability, BIC targets parsimony, and CV targets predictive fit. These criteria are compared empirically in Chapters 6 and 7.

Chapter 5

Simulation Design for the Comparative Study

This chapter describes the simulation framework used to compare the model selection criteria introduced in Chapter 4 for MA-GGMs estimated via penalized Gaussian likelihood. The simulation design follows the construction and conventions in [21], while varying graph topology, sparsity, and sample size to assess selector behavior under controlled conditions. The associated numerical results and discussions are reported in the ensuing chapters.

5.1 Graph Types

We consider three graph families over the node set $V = [p]$ with $p = 100$, representing common structural regimes used in the graphical model literature [2, 7, 10].

Erdős–Rényi (ER) graphs: In an ER graph, each pair of nodes is connected independently with the same probability fracp, so that no node is structurally favored over any other [7, 10]. This produces approximately homogeneous degree distributions in which most nodes have a similar number of connections. In this thesis, we vary $\mathrm{fracp} \in \{0.01, 0.05, 0.10, 0.20\}$, yielding increasing graph density as fracp increases.

Barabási–Albert (BA) graphs: BA graphs are constructed sequentially: starting from a small seed graph, each new node connects to M_0 existing nodes, with the probability of attaching to a given node proportional to its current degree [2, 16]. This preferential attachment mechanism

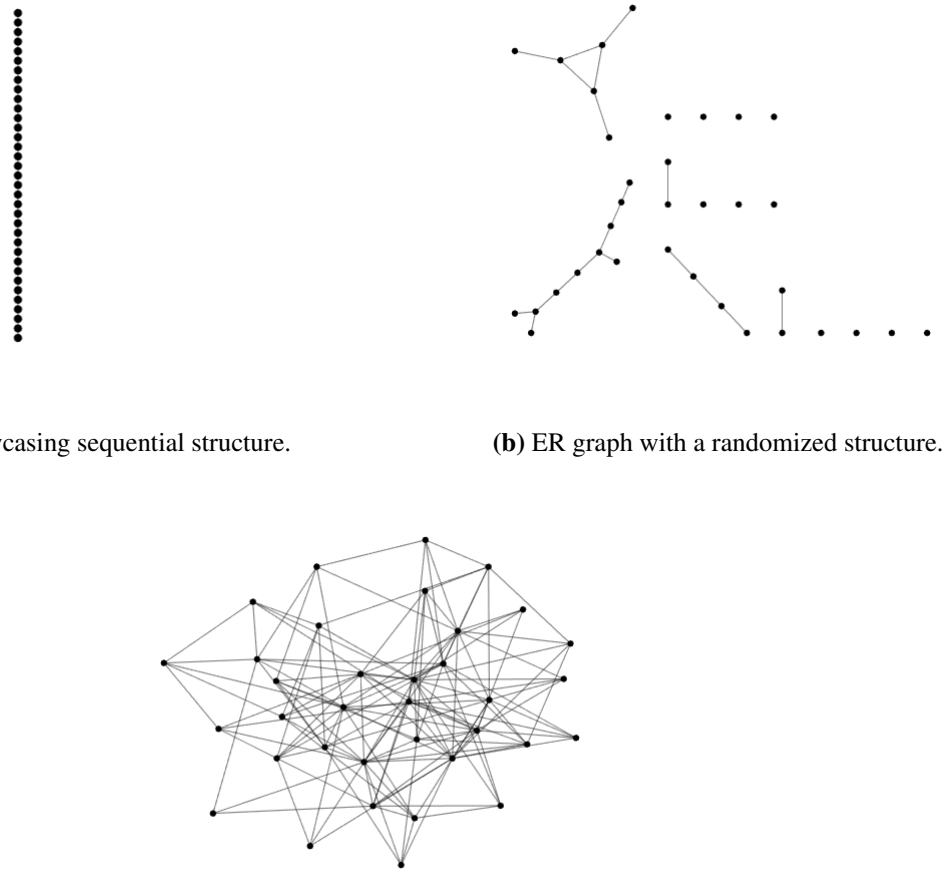
causes well connected nodes to attract even more connections, producing hub-dominated, heavy-tailed degree distributions characteristic of many real-world networks. We vary $M_0 \in \{2, 3, 4, 5\}$.

Chain graphs (CG): Chain graphs connect nodes sequentially, with edges of the form $\{i, i+1\}$ for $i = 1, \ldots, p-1$. This construction yields exactly $p-1$ edges (99 edges for $p = 100$, giving a sparsity factor of approximately 0.02); thus, the sparsity level is fully determined by p and cannot be adjusted independently. As a result, chain graphs do not admit a tunable density parameter and are unsuitable for studying the effect of varying graph density. In this thesis, chain graphs are therefore used only as a baseline topology to illustrate performance under a fixed, low-density structure, while density-dependent analyses are conducted using ER and BA graphs.

For each realized graph, sparsity is summarized by the realized sparsity factor, defined as the fraction of possible unordered node pairs that are connected by an edge:

$$\mathrm{spfac} = \frac{2E}{p(p-1)}, \tag{5.1}$$

where E denotes the number of undirected edges (i.e., the number of unordered pairs $\{i, j\}$ with $i \neq j$ that are connected). For ER graphs, $\mathrm{spfac} = \mathrm{fracp}$ in expectation. For BA graphs, $\mathrm{spfac} \approx 2M_0/(p-1)$ for large p.



(a) Chain graph showcasing sequential structure.

(b) ER graph with a randomized structure.

(c) BA graph exhibiting scale-free structure.

Figure 5.1. Example graphs showcasing the inherent structures of the graph types used in the simulations.

5.2 Data Generation

Each node $j \in [p]$ is associated with an m-dimensional attribute vector $\boldsymbol{z}_j \in \mathbb{R}^m$. Throughout the simulations, we fix $p = 100$ and $m = 4$; thus, the stacked dimension is $d := mp = 400$. Define the stacked random vector $\boldsymbol{x} = [\boldsymbol{z}_1^\top \ \cdots \ \boldsymbol{z}_p^\top]^\top \in \mathbb{R}^d$. Data are generated from a zero-mean Gaussian MA-GGM with precision matrix $\boldsymbol{\Omega} \in \mathbb{R}^{d\times d}$ partitioned into $p \times p$ groups $\boldsymbol{\Omega}^{(ij)} \in \mathbb{R}^{m\times m}$. For each node $j \in [p]$, the diagonal group $\boldsymbol{\Omega}^{(jj)}$ is defined by $[\boldsymbol{\Omega}^{(jj)}]_{st} = 0.5^{|s-t|}$ for $s, t \in [m]$, which induces correlation among attributes within each node. For distinct nodes $i \neq j$ and $s, t \in [m]$, the

off-diagonal group $\mathbf{\Omega}^{(ij)}$ is generated as follows:

$$[\mathbf{\Omega}^{(ij)}]_{st} = \begin{cases} 0, & s = t, \\ U_{st}, & s \neq t \text{ and } \{i,j\} \in \mathcal{E}, \\ 0, & \{i,j\} \notin \mathcal{E}, \end{cases} \tag{5.2}$$

where $\{U_{st}\}_{s,t=1}^{m}$ are sampled independently for each (s,t) from the uniform distribution on $[-0.4,-0.1] \cup [0.1,0.4]$. Here, $\{i,j\} \notin \mathcal{E}$ means $(i,j) \notin \mathcal{E}$ and $(j,i) \notin \mathcal{E}$. If $\{i,j\} \notin \mathcal{E}$, then $\|\mathbf{\Omega}^{(ij)}\|_F = 0$. Symmetry is enforced by setting $\mathbf{\Omega}^{(ji)} = (\mathbf{\Omega}^{(ij)})^\top$. The random generation of off-diagonal entries does not guarantee that $\mathbf{\Omega}$ is positive definite. To ensure positive definiteness, which is required for $\mathbf{\Omega}$ to be a valid precision matrix, we apply the eigenvalue shift used in [21]:

$$\mathbf{\Omega} \leftarrow \mathbf{\Omega} + \big(0.5 - \lambda_{\min}(\mathbf{\Omega})\big)\boldsymbol{I}_d, \tag{5.3}$$

where $\boldsymbol{I}_d$ denotes the $d \times d$ identity matrix (recall $d = mp$). The covariance matrix is then $\mathbf{\Sigma} = \mathbf{\Omega}^{-1}$, and we draw N i.i.d. samples from $\mathcal{N}_d(\mathbf{0}, \mathbf{\Sigma})$. We vary the sample size over $N \in \{200, 400, 800\}$. Additional larger sample sizes $N \in \{1200, 2400, 3200\}$ are included for CV only, across all three graph families, to further assess CV behavior in the regime where validation folds become large enough to stabilize the loss.

5.3 Estimation and Model Selection

Given the sample covariance matrix, the precision matrix is estimated by solving the penalized likelihood problem (3.6) in Chapter 3 using either the lasso penalty (3.2) or LSP (3.3). The mixing parameter is fixed at $\alpha = 0.05$, and for LSP, we set the log-sum tuning parameter to $\epsilon = 10^{-4}$ following [21]. For each dataset, regularization parameters are selected over the grids described in Section 4.1. For SS, we use $B = 20$ subsamples, subsample size $b_N = \lfloor 10\sqrt{N} \rfloor$, instability

threshold $\beta = 0.05$, and pruning threshold $p_{\text{thr}} = 0.60$. For CV, we use $K = 5$ folds. For all three selection criteria, the estimated graph is determined from $\widehat{\boldsymbol{Z}}_\lambda$ as described in Chapter 3.

5.4 Performance Measures

The performance of each selection method is evaluated by comparing the estimated graph to the ground truth conditional independence graph used to generate the data. A true edge between distinct nodes $i \neq j$ is present in the ground truth graph if $\|\boldsymbol{\Omega}^{(ij)}\|_F > 0$, meaning at least one entry of the off-diagonal group is nonzero. A true non-edge corresponds to $\|\boldsymbol{\Omega}^{(ij)}\|_F = 0$, meaning the entire group is zero. An estimated edge is declared present if $\|\widehat{\boldsymbol{Z}}^{(ij)}\|_F > 0$ and absent otherwise. Let TP, FP, and FN denote the numbers of true positives, false positives, and false negatives, respectively. We report the F_1-score (F1) and the Hamming distance (HD):

$$\mathrm{F1} = \frac{2\,\mathrm{TP}}{2\,\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}, \qquad \mathrm{HD} = \mathrm{FP} + \mathrm{FN}. \tag{5.4}$$

The F_1-score summarizes the trade-off between false positives and false negatives, with higher values indicating more accurate recovery of the true edge set. The Hamming distance directly counts the total number of edge selection errors; thus, smaller values correspond to more accurate graph estimation. All reported performance measures are averaged over 50 Monte Carlo trials for each experimental configuration.

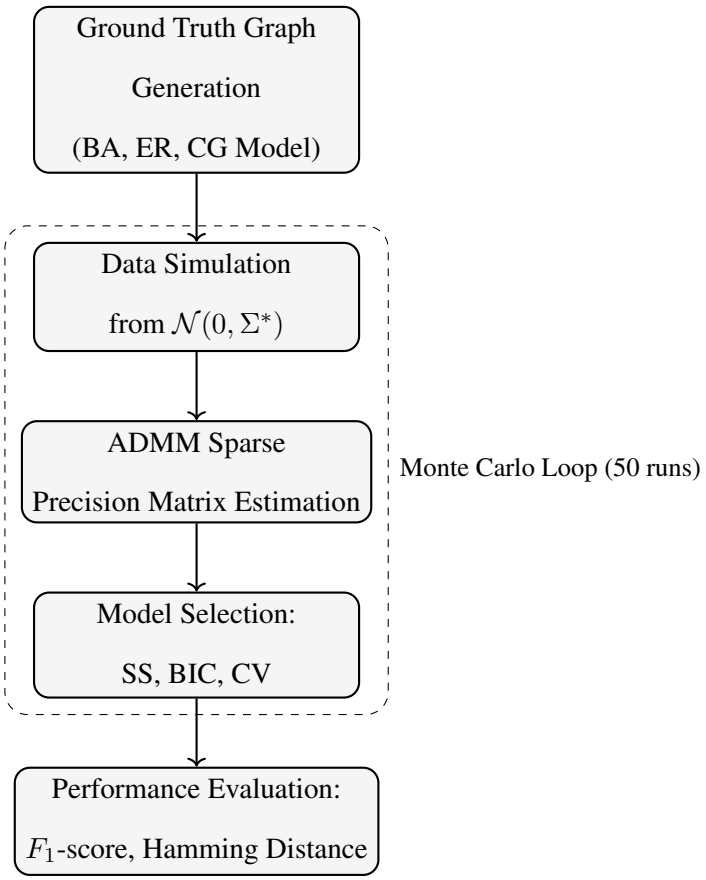


Figure 5.2. Overall experimental pipeline. Ground truth graphs (Barabási–Albert, Erdős–Rényi, or chain graph) $\rightarrow$ Gaussian data simulation $\rightarrow$ ADMM sparse precision estimation (with lasso or log-sum penalties) $\rightarrow$ model selection (stability selection, BIC, CV) $\rightarrow$ performance evaluation (F_1-score, Hamming distance). The dashed group indicates the Monte Carlo loop repeated 50 times.

5.5 Summary

This chapter described the simulation framework, including the three graph families (ER, BA, and chain), the data generation procedure, the estimation and model selection setup, and the performance measures used to evaluate structural recovery. The numerical results of these experiments are presented in the next chapter.

Chapter 6

Results and Comparative Analysis

In this chapter, we report the numerical results of the comparative study. For each configuration, we show the mean with one standard deviation in parentheses over 50 Monte Carlo runs. Performance is summarized by the F_1-score (edge detection quality) and the Hamming distance (total number of edge mismatches). The precision matrix $\mathbf{\Omega}$ is estimated with either lasso or LSP, and the regularization parameter λ is chosen by SS, BIC, or CV. All three selection methods are evaluated at sample sizes $N \in \{200, 400, 800\}$; additional CV-only results for $N \in \{1200, 2400, 3200\}$ are reported separately. Under each graph setting, for ease of comparison, the best result for each sample size (N) is highlighted in bold. Higher F_1-scores correspond to improved edge recovery with an ideal F_1-score being 1, whereas lower Hamming distances indicate fewer structural errors with 0 being the best possible Hamming distance.

6.1 Overall Performance at Matched Sparsity

We first compare SS, BIC, and CV on BA, ER, and chain graphs under comparable sparsity regimes (BA with $M_0 = 2$, ER with $\mathrm{fracp} = 0.01$, and chain graphs with $\mathrm{spfac} = 0.02$). Matching sparsity levels across graph families provides a controlled basis for comparison, ensuring that observed differences in performance are not driven primarily by graph density. This allows the behavior of each selection method to be assessed more directly as a function of graph topology, sample size, and penalty choice at a fixed level of sparsity. Across these settings, we observe that: BIC with

LSP achieves the highest F_1-score and lowest Hamming distance once N is moderate or large; SS (especially with LSP and pruning) performs well at small or moderate N; and CV performs poorly at $N \in \{200, 400, 800\}$ but improves as N increases.

Table 6.1. Performance on Barabási–Albert (BA) graphs (p=100, m=4, M_0=2, spfac≈0.04). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

Method	N=200		N=400		N=800	
	F1	HD	F1	HD	F1	HD
SS (Lasso) — Unpruned	0.60 (0.06)	124.30 (17.21)	0.70 (0.08)	92.70 (18.60)	0.80 (0.07)	64.40 (18.30)
SS (Lasso) — Pruned	0.59 (0.05)	151.10 (17.60)	0.74 (0.06)	95.00 (20.60)	0.85 (0.05)	59.70 (19.60)
SS (LSP) — Unpruned	0.62 (0.07)	**109.20** (15.00)	0.73 (0.10)	80.90 (21.50)	0.85 (0.07)	47.80 (20.00)
SS (LSP) — Pruned	**0.64** (0.07)	111.80 (16.10)	**0.81** (0.08)	**61.00** (20.80)	0.94 (0.04)	21.30 (11.80)
BIC (Lasso)	0.52 (0.08)	274.00 (128.20)	0.75 (0.10)	79.70 (23.60)	0.92 (0.05)	29.20 (15.60)
BIC (LSP)	**0.64** (0.07)	141.00 (37.30)	0.80 (0.07)	68.00 (17.70)	**0.97** (0.03)	**8.40** (10.00)
CV (Lasso)	0.08 (0.002)	4481.50 (117.5)	0.09 (0.01)	4044.10 (256.5)	0.09 (0.01)	3750.70 (334.5)
CV (LSP)	0.09 (0.01)	3661.5 (237.2)	0.14 (0.02)	2327.1 (383.6)	0.30 (0.09)	978.5 (373.2)

On BA graphs (Table 6.1), CV produces F_1-scores below 0.10 with Hamming distances in the thousands at $N \leq 800$, indicating that it selects graphs far denser than the truth. In contrast, SS with LSP (pruned) leads at $N = 200$ and $N = 400$, while BIC with LSP dominates at $N = 800$ with an F_1 of 0.97. The high variance of BIC (Lasso) at $N = 200$ (HD std of 128.20) reflects its sensitivity to the grid when sample size is small.

Table 6.2. CV on Barabási–Albert (BA) graphs at larger N (p=100, m=4, M_0=2, spfac≈0.04). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

Method	N=1200		N=2400		N=3200	
	F1	HD	F1	HD	F1	HD
CV (Lasso)	0.14 (0.04)	2525.10 (550.80)	0.54 (0.13)	369.40 (204.20)	0.77 (0.10)	121.60 (67.60)
CV (LSP)	**0.78** (0.13)	**124.22** (107.82)	**1.00** (0.001)	**0.06** (0.31)	**1.00** (0.00)	**0.00** (0.00)

At larger sample sizes (Table 6.2), CV with LSP recovers sharply, reaching near-perfect recovery by $N = 2400$. CV with lasso remains substantially worse, confirming the benefit of the non-convex penalty even for CV-based selection.

Table 6.3. Performance on Erdős–Rényi (ER) graphs (p=100, m=4, fracp=0.01, spfac≈0.01). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

Method	N=200		N=400		N=800	
	F1	HD	F1	HD	F1	HD
SS (Lasso) — Unpruned	0.83 (0.04)	18.30 (5.20)	0.98 (0.01)	1.66 (1.50)	0.99 (0.01)	0.32 (0.60)
SS (Lasso) — Pruned	0.65 (0.05)	49.50 (8.30)	0.93 (0.02)	8.00 (2.70)	0.97 (0.02)	2.70 (1.70)
SS (LSP) — Unpruned	**0.88** (0.04)	**12.90** (4.76)	**0.99** (0.01)	**0.70** (1.01)	**0.99** (0.003)	**0.08** (0.27)
SS (LSP) — Pruned	0.76 (0.04)	29.38 (5.97)	0.98 (0.01)	2.38 (1.40)	**0.99** (0.004)	0.18 (0.43)
BIC (Lasso)	0.32 (0.14)	275.80 (169.60)	0.95 (0.06)	5.70 (9.20)	0.96 (0.13)	16.40 (101.80)
BIC (LSP)	0.86 (0.09)	16.96 (14.68)	0.98 (0.02)	1.56 (1.92)	**0.99** (0.01)	0.98 (1.38)
CV (Lasso)	0.02 (0.003)	4659.0 (90.5)	0.02 (0.003)	4189.4 (236.1)	0.02 (0.01)	3993.6 (473.8)
CV (LSP)	0.03 (0.01)	3416.5 (392.7)	0.10 (0.04)	1057.4 (394.9)	0.41 (0.24)	349.9 (566.5)

On sparse ER graphs (Table 6.3), all methods except CV perform well even at moderate N. SS with LSP (unpruned) achieves $F_1 \geq 0.88$ at $N = 200$ and near-perfect recovery by $N = 400$. BIC with lasso again shows high variance at $N = 200$ (HD std of 169.60), while BIC with LSP is competitive throughout. CV remains poor across all sample sizes in this range.

Table 6.4. CV on Erdős–Rényi (ER) graphs at larger N (p=100, m=4, fracp=0.01, spfac≈0.01). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

Method	N=1200		N=2400		N=3200	
	F1	HD	F1	HD	F1	HD
CV (Lasso)	0.04 (0.01)	2421.20 (688.90)	0.30 (0.18)	454.30 (635.20)	0.66 (0.23)	173.70 (566.20)
CV (LSP)	**0.94** (0.15)	**17.34** (80.19)	**1.00** (0.002)	**0.02** (0.14)	**1.00** (0.00)	**0.00** (0.00)

Table 6.5. Performance on chain graphs (p=100, m=4, spfac=0.02). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

Method	N=200		N=400		N=800	
	F1	HD	F1	HD	F1	HD
SS (Lasso) — Unpruned	0.90 (0.02)	19.84 (4.56)	0.98 (0.01)	2.72 (1.97)	0.99 (0.00)	0.22 (0.50)
SS (Lasso) — Pruned	0.81 (0.02)	44.24 (6.31)	0.97 (0.01)	6.52 (2.51)	0.99 (0.00)	1.26 (0.77)
SS (LSP) — Unpruned	0.93 (0.02)	13.44 (4.30)	**0.99** (0.00)	**0.58** (0.78)	**1.00** (0.00)	**0.00** (0.00)
SS (LSP) — Pruned	0.87 (0.02)	29.66 (5.68)	0.98 (0.01)	2.42 (1.30)	0.99 (0.00)	0.08 (0.27)
BIC (Lasso)	0.68 (0.10)	99.74 (56.86)	0.98 (0.02)	3.60 (3.42)	0.95 (0.03)	10.10 (7.73)
BIC (LSP)	**0.95** (0.02)	**9.14** (3.70)	0.98 (0.02)	3.36 (4.40)	0.99 (0.00)	0.60 (1.34)
CV (Lasso)	0.04 (0.00)	4453.6 (85.4)	0.05 (0.00)	3690.2 (272.5)	0.07 (0.01)	2709.2 (443.4)
CV (LSP)	0.07 (0.01)	2607.6 (297.8)	0.36 (0.11)	413.0 (238.5)	0.95 (0.05)	10.5 (12.3)

Chain graphs (Table 6.5) are the easiest topology in our experiments. SS with LSP achieves perfect recovery by $N = 800$. Notably, CV with LSP reaches $F_1 = 0.95$ at $N = 800$ on chain graphs, a setting where it still fails on the other topologies, suggesting that the simpler structure of chain graphs partially compensates for CV's tendency to overselect edges.

Table 6.6. CV on chain graphs at larger N (p=100, m=4, spfac=0.02). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

Method	N=1200		N=2400		N=3200	
	F1	HD	F1	HD	F1	HD
CV (Lasso)	0.17 (0.04)	1081.3 (327.7)	0.87 (0.08)	32.5 (24.3)	0.98 (0.01)	3.3 (1.8)
CV (LSP)	**1.00** (0.00)	**0.00** (0.00)	**1.00** (0.00)	**0.00** (0.00)	**1.00** (0.00)	**0.00** (0.00)

6.2 Impact of Graph Density

We next examine how graph density affects model selection performance. As density increases, the number of true edges grows and the opportunity for false discoveries rises, making structural recovery harder. By varying density while holding other factors fixed, we isolate density effects within each graph family without confounding them with changes in topology.

Density variation is studied using ER and BA graphs, which admit explicit tuning parameters. For BA graphs, density increases with the attachment parameter M_0, while for ER graphs it is controlled by the connection probability fracp. Chain graphs do not allow independent control of density and are therefore excluded from this analysis. We report F_1-scores in compact form and present Hamming distance results separately for readability. Results for larger sample sizes under CV are again shown at the end of each group. Across all topologies, F_1-score decreases and Hamming distance increases as density grows. Pruning is most helpful for recovery in sparser graphs. At matched or similar levels of sparsity, BA graphs remain harder to recover than both ER and chain graphs.

Table 6.7. F_1-score on Barabási–Albert (BA) graphs across M_0 and spfac (p=100, m=4). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs. SS/BIC/CV at N=200, 400, 800.

	Lasso				LSP			
Method	M_0=2 (0.04)	3 (0.06)	4 (0.08)	5 (0.10)	2 (0.04)	3 (0.06)	4 (0.08)	5 (0.10)
N=200								
SS (Unpruned)	0.60 (0.06)	0.56 (0.05)	0.51 (0.04)	0.46 (0.04)	0.62 (0.07)	0.55 (0.07)	0.48 (0.05)	0.40 (0.04)
SS (Pruned)	0.59 (0.05)	0.57 (0.04)	0.56 (0.04)	0.52 (0.03)	**0.64** (0.07)	0.57 (0.06)	0.51 (0.05)	0.44 (0.04)
BIC	0.52 (0.08)	0.57 (0.05)	0.59 (0.04)	**0.57** (0.05)	**0.64** (0.07)	**0.63** (0.05)	**0.61** (0.04)	**0.57** (0.04)
CV	0.08 (0.002)	0.11 (0.002)	0.15 (0.004)	0.18 (0.004)	0.09 (0.01)	0.14 (0.01)	0.17 (0.01)	0.20 (0.01)
N=400								
SS (Unpruned)	0.69 (0.08)	0.60 (0.06)	0.53 (0.05)	0.49 (0.04)	0.73 (0.10)	0.62 (0.07)	0.53 (0.07)	0.46 (0.06)
SS (Pruned)	0.74 (0.06)	0.69 (0.04)	0.64 (0.04)	0.60 (0.04)	**0.81** (0.08)	0.72 (0.06)	0.63 (0.06)	0.56 (0.06)
BIC	0.75 (0.10)	0.71 (0.09)	0.67 (0.09)	**0.66** (0.07)	0.80 (0.07)	**0.73** (0.07)	**0.69** (0.08)	0.65 (0.06)
CV	0.09 (0.01)	0.13 (0.01)	0.16 (0.01)	0.19 (0.01)	0.14 (0.02)	0.20 (0.03)	0.24 (0.02)	0.28 (0.03)
N=800								
SS (Unpruned)	0.80 (0.07)	0.72 (0.05)	0.63 (0.05)	0.57 (0.05)	0.85 (0.07)	0.78 (0.05)	0.68 (0.08)	0.58 (0.07)
SS (Pruned)	0.85 (0.05)	0.83 (0.03)	0.77 (0.04)	0.71 (0.04)	0.94 (0.04)	0.89 (0.03)	0.83 (0.06)	0.74 (0.06)
BIC	0.92 (0.05)	0.89 (0.04)	0.84 (0.05)	0.80 (0.06)	**0.97** (0.03)	**0.94** (0.07)	**0.88** (0.08)	**0.81** (0.06)
CV	0.09 (0.01)	0.13 (0.01)	0.18 (0.02)	0.22 (0.02)	0.30 (0.09)	0.38 (0.07)	0.45 (0.09)	0.49 (0.07)

Table 6.8. Hamming distance on Barabási–Albert (BA) graphs across M_0 and spfac using lasso (p=100, m=4). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs. SS/BIC/CV at N=200, 400, 800.

	Lasso			
Method	M_0=2 (0.04)	3 (0.06)	4 (0.08)	5 (0.10)
N=200				
SS (Unpruned)	124.30 (17.20)	191.40 (16.90)	257.40 (13.80)	336.40 (15.50)
SS (Pruned)	151.10 (17.60)	211.70 (18.30)	266.10 (16.00)	335.70 (17.10)
BIC	274.00 (128.20)	268.90 (70.10)	322.60 (62.40)	387.30 (108.00)
CV	4481.50 (117.5)	4344.8 (112.8)	4267.0 (105.6)	4170.3 (117.8)
N=400				
SS (Unpruned)	92.70 (18.60)	161.70 (17.50)	237.00 (19.30)	307.40 (17.50)
SS (Pruned)	95.00 (20.60)	148.60 (18.30)	210.20 (19.30)	271.10 (19.10)
BIC	79.70 (23.60)	130.10 (28.00)	190.80 (33.80)	238.80 (33.60)
CV	4044.10 (256.5)	3921.0 (259.4)	3860.6 (218.7)	3772.6 (204.5)
N=800				
SS (Unpruned)	64.40 (18.30)	122.70 (16.90)	196.90 (20.80)	272.00 (23.10)
SS (Pruned)	56.70 (19.60)	91.30 (16.50)	144.10 (22.10)	210.90 (22.90)
BIC	29.20 (15.60)	58.50 (19.60)	100.70 (28.80)	152.70 (36.60)
CV	3750.70 (334.5)	3627.9 (325.0)	3369.0 (359.5)	3297.2 (292.2)

Table 6.9. Hamming distance on Barabási–Albert (BA) graphs across M_0 and spfac using LSP (p=100, m=4). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs. SS/BIC/CV at N=200, 400, 800.

	LSP			
Method	M_0=2 (0.04)	3 (0.06)	4 (0.08)	5 (0.10)
N=200				
SS (Unpruned)	**109.20** (15.00)	180.80 (17.74)	255.60 (15.86)	342.46 (16.34)
SS (Pruned)	111.80 (16.10)	**179.42** (18.43)	250.80 (15.90)	334.60 (17.18)
BIC	141.00 (37.30)	187.22 (30.56)	**248.50** (27.70)	**330.38** (48.53)
CV	3661.5 (237.2)	3507.0 (189.4)	3504.2 (153.8)	3460.4 (213.5)
N=400				
SS (Unpruned)	80.90 (21.50)	152.38 (20.41)	234.74 (23.86)	315.36 (24.11)
SS (Pruned)	**61.00** (20.80)	121.36 (20.52)	196.50 (25.06)	274.90 (25.30)
BIC	68.00 (17.70)	**116.70** (22.88)	**172.56** (30.35)	**230.30** (31.66)
CV	2327.1 (383.6)	2329.1 (371.6)	2379.5 (317.6)	2346.1 (303.2)
N=800				
SS (Unpruned)	47.80 (20.00)	99.42 (20.10)	172.70 (33.86)	263.36 (32.90)
SS (Pruned)	21.30 (11.80)	53.20 (14.85)	106.64 (32.23)	186.66 (35.03)
BIC	**8.40** (10.00)	**28.12** (30.71)	**73.12** (48.51)	**143.14** (37.84)
CV	978.5 (373.2)	954.9 (259.2)	973.2 (362.0)	996.8 (298.1)

Table 6.10. CV F_1-score on Barabási–Albert (BA) graphs across M_0 at larger N (p=100, m=4). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

	F1			
Method / N	M_0=2 (0.04)	3 (0.06)	4 (0.08)	5 (0.10)
N=1200				
CV (Lasso)	0.14 (0.04)	0.21 (0.05)	0.26 (0.04)	0.31 (0.06)
CV (LSP)	**0.78** (0.13)	**0.83** (0.09)	**0.84** (0.08)	**0.84** (0.08)
N=2400				
CV (Lasso)	0.54 (0.13)	0.66 (0.12)	0.71 (0.10)	0.74 (0.06)
CV (LSP)	**1.00** (0.001)	**1.00** (0.001)	**1.00** (0.00)	**1.00** (0.001)
N=3200				
CV (Lasso)	0.77 (0.10)	0.86 (0.05)	0.87 (0.05)	0.89 (0.04)
CV (LSP)	**1.00** (0.00)	**1.00** (0.00)	**1.00** (0.00)	**1.00** (0.00)

Table 6.11. CV Hamming distance on Barabási–Albert (BA) graphs across M_0 at larger N (p=100, m=4). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

	HD			
Method / N	M_0=2 (0.04)	3 (0.06)	4 (0.08)	5 (0.10)
N=1200				
CV (Lasso)	2525.10 (550.80)	2188.60 (578.50)	2178.10 (478.30)	2073.70 (493.10)
CV (LSP)	**124.22** (107.82)	**120.16** (82.67)	**146.80** (91.63)	**182.66** (105.53)
N=2400				
CV (Lasso)	369.40 (204.20)	319.10 (182.80)	320.20 (162.20)	328.30 (108.70)
CV (LSP)	**0.06** (0.31)	**0.12** (0.43)	**0.08** (0.27)	**0.24** (0.65)
N=3200				
CV (Lasso)	121.60 (67.60)	91.80 (43.90)	111.50 (51.20)	111.70 (56.10)
CV (LSP)	**0.00** (0.00)	**0.02** (0.14)	**0.06** (0.24)	**0.04** (0.20)

Table 6.12. F_1-score on Erdős–Rényi (ER) graphs across fracp (p=100, m=4). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs. SS/BIC/CV at N=200, 400, 800.

	Lasso				LSP			
Method	fracp=0.01	0.05	0.10	0.20	fracp=0.01	0.05	0.10	0.20
N=200								
SS (Unpruned)	0.83 (0.04)	0.73 (0.05)	0.51 (0.04)	0.21 (0.03)	**0.88** (0.04)	0.80 (0.05)	0.55 (0.04)	0.48 (0.05)
SS (Pruned)	0.65 (0.05)	0.75 (0.04)	0.58 (0.04)	0.30 (0.03)	0.76 (0.04)	0.80 (0.04)	0.58 (0.04)	0.51 (0.05)
BIC	0.32 (0.14)	0.52 (0.06)	0.57 (0.05)	0.59 (0.02)	0.86 (0.09)	**0.81** (0.03)	**0.72** (0.03)	**0.61** (0.04)
CV	0.02 (0.003)	0.10 (0.01)	0.19 (0.01)	0.34 (0.01)	0.03 (0.01)	0.13 (0.01)	0.23 (0.01)	0.38 (0.01)
N=400								
SS (Unpruned)	0.98 (0.01)	0.85 (0.03)	0.57 (0.06)	0.16 (0.03)	**0.99** (0.01)	0.91 (0.03)	0.68 (0.07)	0.53 (0.07)
SS (Pruned)	0.93 (0.02)	0.90 (0.02)	0.72 (0.04)	0.29 (0.03)	0.98 (0.01)	**0.95** (0.02)	0.76 (0.06)	0.63 (0.06)
BIC	0.95 (0.06)	0.93 (0.01)	**0.89** (0.02)	**0.80** (0.01)	0.98 (0.02)	0.94 (0.04)	0.83 (0.03)	0.70 (0.08)
CV	0.02 (0.003)	0.11 (0.01)	0.19 (0.01)	0.35 (0.01)	0.10 (0.04)	0.22 (0.03)	0.31 (0.03)	0.45 (0.02)
N=800								
SS (Unpruned)	**0.99** (0.01)	0.94 (0.02)	0.76 (0.06)	0.17 (0.03)	**0.99** (0.003)	0.98 (0.01)	0.86 (0.05)	0.69 (0.08)
SS (Pruned)	0.97 (0.02)	0.97 (0.01)	0.87 (0.03)	0.31 (0.05)	**0.99** (0.004)	**0.99** (0.004)	0.93 (0.03)	0.83 (0.06)
BIC	0.96 (0.13)	0.98 (0.02)	0.97 (0.01)	**0.92** (0.01)	**0.99** (0.01)	0.98 (0.01)	**0.98** (0.01)	0.88 (0.08)
CV	0.02 (0.01)	0.12 (0.01)	0.21 (0.01)	0.36 (0.01)	0.41 (0.24)	0.66 (0.10)	0.63 (0.06)	0.63 (0.04)

Table 6.13. Hamming distance on Erdős–Rényi (ER) graphs across fracp using lasso (p=100, m=4). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs. SS/BIC/CV at N=200, 400, 800.

	Lasso			
Method	fracp=0.01	0.05	0.10	0.20
N=200				
SS (Unpruned)	18.30 (5.20)	112.30 (19.00)	334.00 (26.80)	881.90 (35.40)
SS (Pruned)	49.50 (8.30)	118.00 (16.10)	316.00 (30.00)	834.60 (35.70)
BIC	275.80 (169.60)	442.00 (116.20)	694.90 (142.40)	1167.90 (131.50)
CV	4659.0 (90.5)	4485.5 (41.6)	4269.2 (45.40)	3830.0 (38.2)
N=400				
SS (Unpruned)	1.66 (1.50)	64.00 (14.80)	289.30 (34.10)	908.40 (32.90)
SS (Pruned)	8.00 (2.70)	45.20 (8.90)	218.50 (30.20)	828.10 (38.00)
BIC	5.70 (9.20)	35.10 (8.00)	**129.00** (21.00)	434.60 (47.20)
CV	4189.4 (236.1)	4171.1 (108.7)	4088.3 (88.10)	3752.2 (58.0)
N=800				
SS (Unpruned)	0.32 (0.60)	26.00 (9.50)	197.20 (34.10)	903.50 (37.90)
SS (Pruned)	2.70 (1.70)	13.30 (5.60)	109.00 (27.30)	808.80 (46.70)
BIC	16.40 (101.80)	8.40 (9.00)	30.00 (6.20)	151.50 (14.50)
CV	3993.60 (473.8)	3532.0 (230.70)	3595.7 (139.20)	3504.4 (80.6)

Table 6.14. Hamming distance on Erdős–Rényi (ER) graphs across fracp using LSP (p=100, m=4). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs. SS/BIC/CV at N=200, 400, 800.

	LSP			
Method	fracp=0.01	0.05	0.10	0.20
N=200				
SS (Unpruned)	**12.90** (4.76)	89.96 (18.84)	311.00 (29.50)	255.56 (15.87)
SS (Pruned)	29.38 (5.97)	90.88 (17.14)	297.80 (28.20)	250.80 (15.94)
BIC	16.96 (14.68)	**89.04** (19.05)	**245.90** (30.00)	**248.48** (27.70)
CV	3416.5 (392.7)	3390.1 (165.3)	3334.8 (165.4)	3116.7 (105.0)
N=400				
SS (Unpruned)	**0.70** (1.01)	39.36 (12.42)	239.80 (44.20)	234.74 (23.87)
SS (Pruned)	2.38 (1.40)	**24.82** (8.15)	190.60 (42.60)	196.50 (25.06)
BIC	1.56 (1.92)	28.60 (16.10)	139.90 (22.50)	**172.56** (30.35)
CV	1057.4 (394.9)	1761.9 (307.9)	2221.4 (280.7)	2423.4 (179.8)
N=800				
SS (Unpruned)	**0.08** (0.27)	9.72 (6.11)	113.50 (36.70)	172.70 (33.87)
SS (Pruned)	0.18 (0.43)	**2.66** (2.18)	57.10 (24.10)	106.64 (32.23)
BIC	0.98 (1.38)	8.24 (6.68)	**17.20** (13.40)	**73.12** (48.51)
CV	349.9 (566.5)	276.7 (136.5)	578.6 (165.1)	1197.9 (195.9)

Table 6.15. CV F_1-score on Erdős–Rényi (ER) graphs across fracp at larger N (p=100, m=4). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

	F1			
Method / N	fracp=0.01	0.05	0.10	0.20
N=1200				
CV (Lasso)	0.04 (0.01)	0.18 (0.02)	0.27 (0.02)	0.40 (0.02)
CV (LSP)	**0.94** (0.15)	**0.98** (0.02)	**0.95** (0.02)	**0.89** (0.03)
N=2400				
CV (Lasso)	0.30 (0.18)	0.65 (0.08)	0.58 (0.05)	0.60 (0.04)
CV (LSP)	**1.00** (0.002)	**1.00** (0.00)	**1.00** (0.00)	**1.00** (0.001)
N=3200				
CV (Lasso)	0.66 (0.23)	0.86 (0.05)	0.79 (0.04)	0.73 (0.03)
CV (LSP)	**1.00** (0.00)	**1.00** (0.00)	**1.00** (0.00)	**1.00** (0.00)

Table 6.16. CV Hamming distance on Erdős–Rényi (ER) graphs across fracp at larger N (p=100, m=4). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

	Hamming Distance			
Method / N	fracp=0.01	0.05	0.10	0.20
N=1200				
CV (Lasso)	2421.20 (688.90)	2232.70 (324.80)	2730.40 (227.60)	2918.30 (172.40)
CV (LSP)	**17.34** (80.19)	**8.86** (8.05)	**53.74** (26.01)	**250.30** (75.72)
N=2400				
CV (Lasso)	454.30 (635.20)	273.10 (96.40)	746.50 (182.90)	1333.50 (197.80)
CV (LSP)	**0.02** (0.14)	**0.00** (0.00)	**0.00** (0.00)	**0.60** (0.96)
N=3200				
CV (Lasso)	173.70 (566.20)	82.10 (36.60)	271.00 (68.60)	750.70 (132.30)
CV (LSP)	**0.00** (0.00)	**0.00** (0.00)	**0.00** (0.00)	**0.04** (0.20)

6.3 Stability Selection and the Role of β

Finally, we examine the role of the instability threshold β in SS. The parameter β directly controls the tolerated level of selection variability across subsamples and therefore acts as a sparsity control knob; smaller values of β favor more conservative, stable graphs, while larger values permit denser solutions with potentially higher recall. By varying $\beta \in \{0.05, 0.075, 0.10\}$, we assess the sensitivity of SS to this tuning choice and evaluate the trade-off between false positives and false

negatives. This analysis helps clarify how β affects structural recovery under different sample sizes and graph families. We report F_1-scores and Hamming distances for BA, ER, and chain graphs to summarize these effects.

Table 6.17. SS on BA across β using lasso/LSP (p=100, m=4, M_0=2, spfac$\approx$0.04). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

Method	**F1**			**HD**		
	N=200	N=400	N=800	N=200	N=400	N=800
Lasso (β=0.05)						
Unpruned	0.60 (0.06)	0.70 (0.09)	0.82 (0.05)	125.5 (21.2)	90.5 (20.8)	61.5 (16.4)
Pruned	0.59 (0.05)	0.74 (0.07)	0.85 (0.05)	151.9 (19.6)	92.1 (20.9)	59.4 (20.5)
Lasso (β=0.075)						
Unpruned	0.62 (0.07)	0.75 (0.08)	0.84 (0.05)	133.1 (18.0)	80.8 (22.0)	49.3 (17.3)
Pruned	0.58 (0.04)	0.76 (0.05)	0.84 (0.05)	190.1 (18.2)	99.1 (16.7)	66.0 (20.7)
Lasso (β=0.10)						
Unpruned	0.62 (0.05)	0.76 (0.07)	0.87 (0.05)	144.4 (16.4)	76.9 (19.1)	43.7 (16.2)
Pruned	0.54 (0.03)	0.73 (0.04)	0.83 (0.04)	238.8 (20.9)	116.0 (14.7)	74.5 (17.0)
LSP (β=0.05)						
Unpruned	0.62 (0.07)	0.73 (0.10)	0.85 (0.07)	109.2 (15.0)	80.9 (21.5)	47.8 (20.0)
Pruned	0.64 (0.07)	0.81 (0.08)	0.94 (0.04)	111.8 (16.1)	61.0 (20.8)	21.3 (11.8)
LSP (β=0.075)						
Unpruned	**0.66** (0.08)	0.82 (0.06)	0.91 (0.06)	**95.0** (15.01)	57.0 (17.00)	35.0 (13.00)
Pruned	0.65 (0.07)	**0.88** (0.05)	0.96 (0.04)	145.0 (16.0)	75.0 (15.02)	50.0 (17.30)
LSP (β=0.10)						
Unpruned	**0.66** (0.06)	0.81 (0.07)	0.91 (0.05)	118.08 (16.95)	58.96 (19.42)	30.70 (14.24)
Pruned	0.62 (0.05)	0.86 (0.05)	**0.97** (0.02)	155.72 (17.50)	**48.30** (16.45)	**11.08** (6.26)

Table 6.18. SS on ER across β using lasso/LSP (p=100, m=4, fracp=0.01, spfac≈0.01). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

Method	**F1**			**HD**		
	N=200	N=400	N=800	N=200	N=400	N=800
Lasso (β=0.05)						
Unpruned	0.83 (0.04)	0.98 (0.01)	**0.99** (0.01)	18.26 (5.20)	1.66 (1.50)	0.32 (0.58)
Pruned	0.65 (0.05)	0.93 (0.02)	0.97 (0.02)	49.50 (8.27)	8.02 (2.72)	2.74 (1.68)
Lasso (β=0.075)						
Unpruned	0.74 (0.04)	0.98 (0.01)	**0.99** (0.005)	31.64 (6.35)	1.66 (1.26)	0.30 (0.54)
Pruned	0.51 (0.04)	0.85 (0.03)	0.95 (0.02)	90.96 (9.77)	17.94 (4.86)	4.96 (2.55)
Lasso (β=0.10)						
Unpruned	0.64 (0.05)	0.98 (0.01)	**0.99** (0.005)	52.02 (9.88)	2.40 (1.59)	0.30 (0.61)
Pruned	0.39 (0.04)	0.76 (0.04)	0.91 (0.03)	149.20 (15.75)	31.90 (6.27)	9.34 (3.76)
LSP (β=0.05)						
Unpruned	**0.88** (0.04)	**0.99** (0.01)	**0.99** (0.003)	**12.90** (4.76)	0.70 (1.01)	0.08 (0.27)
Pruned	0.76 (0.04)	0.98 (0.01)	**0.99** (0.004)	29.38 (5.97)	2.38 (1.40)	0.18 (0.43)
LSP (β=0.075)						
Unpruned	0.79 (0.04)	**0.99** (0.01)	**0.99** (0.001)	23.62 (5.57)	**0.38** (0.63)	0.06 (0.24)
Pruned	0.61 (0.04)	0.95 (0.02)	**0.99** (0.01)	58.86 (7.45)	5.74 (2.60)	0.48 (0.61)
LSP (β=0.10)						
Unpruned	0.71 (0.04)	**0.99** (0.001)	**0.99** (0.00)	38.76 (6.74)	0.50 (0.70)	**0.04** (0.20)
Pruned	0.50 (0.05)	0.89 (0.03)	0.98 (0.01)	99.04 (10.48)	12.20 (3.23)	1.24 (1.16)

Table 6.19. SS on CG across β using lasso/LSP (p=100, m=4, spfac=0.02). We show the mean with one standard deviation in parentheses over 50 Monte Carlo runs.

Method	**F1**			**HD**		
	N=200	N=400	N=800	N=200	N=400	N=800
Lasso (β=0.05)						
Unpruned	0.90(0.02)	0.98(0.01)	0.99(0.00)	19.84(4.57)	2.72(1.97)	0.22(0.50)
Pruned	0.81(0.02)	0.97(0.01)	0.99(0.00)	44.24(6.31)	6.52(2.51)	1.26(0.77)
Lasso (β=0.075)						
Unpruned	0.86(0.02)	0.98(0.01)	0.99(0.00)	31.18(5.99)	2.34(1.70)	0.16(0.37)
Pruned	0.70(0.03)	0.94(0.01)	0.99(0.01)	81.10(9.51)	12.76(3.00)	2.64(1.38)
Lasso (β=0.10)						
Unpruned	0.80(0.02)	0.98(0.01)	0.99(0.00)	48.22(6.28)	2.68(1.73)	0.12(0.38)
Pruned	0.60(0.02)	0.89(0.02)	0.98(0.01)	131.58(11.88)	23.30(3.85)	4.54(1.95)
LSP (β=0.05)						
Unpruned	**0.93**(0.02)	**0.99**(0.00)	**1.00**(0.00)	**13.44**(4.30)	0.58(0.78)	**0.00**(0.00)
Pruned	0.87(0.02)	0.98(0.00)	0.99(0.00)	29.66(5.68)	2.42(1.30)	0.08 (0.27)
LSP (β=0.075)						
Unpruned	0.89(0.02)	**0.99**(0.00)	**1.00**(0.00)	23.10(4.92)	0.46(0.70)	**0.00**(0.00)
Pruned	0.77(0.024)	0.97(0.01)	0.99(0.00)	57.20(7.79)	5.76(2.41)	0.48(0.67)
LSP (β=0.10)						
Unpruned	0.84(0.02)	**0.99**(0.00)	**1.00**(0.00)	36.48(5.95)	**0.44**(0.61)	**0.00**(0.00)
Pruned	0.67(0.02)	0.94(0.02)	0.99(0.01)	96.30(8.81)	11.70(3.90)	1.12(1.10)

Overall, the results indicate that a larger β tends to increase recall for unpruned graphs (yielding denser estimates and lower HD when N is large enough), while pruning strongly reduces false positives. The trade-off is most visible on BA graphs (Table 6.17), where $\beta = 0.10$ with LSP and pruning achieves the best HD of 11.08 at $N = 800$, compared to 21.3 for $\beta = 0.05$. On ER and chain graphs, the differences across β values are smaller once $N \geq 400$. In practice, $\beta = 0.05$ provides a safe conservative default that avoids overselection, while $\beta \in \{0.075, 0.10\}$ may be preferable when higher recall is desired and pruning is applied.

6.4 Summary

In this chapter, we presented the numerical results for all configurations. A detailed interpretation, practical recommendations, and possible extensions are deferred to Chapter 8. The three main empirical patterns are as follows. BIC with LSP provides the best overall balance between accuracy and sparsity, achieving the highest F_1-scores at $N = 800$ across all topologies. SS with LSP and pruning is highly effective in sparse or small-N regimes, often leading at $N \in \{200, 400\}$. CV is unreliable at $N \leq 800$ but becomes competitive once N reaches approximately 1200–2400, depending on graph topology and penalty choice, with LSP consistently outperforming lasso. In the next chapter, we apply the same methodology to a real-world financial dataset to assess whether these patterns hold outside the controlled simulation setting.

Chapter 7

Real-World Data: Financial Time Series

We apply our methodology to daily financial time series from the S&P 100 index. The dataset consists of 97 stocks observed from April 1, 2015 to April 1, 2020, yielding 1259 trading days. The data were obtained from Yahoo Finance [23] by [21], who provided the dataset for this study. The construction follows [21]. After computing log-returns, which require consecutive pairs of observations, we have $N = 1258$ observations to analyze. For each stock, four daily attributes are considered: high price, low price, close-of-the-day price, and daily trading volume, giving $m = 4$ and $p = 97$. Log-returns are computed as $\ln\big(z_{i\ell}(t)/z_{i\ell}(t-1)\big)$ for the ℓth attribute of the ith stock on day t, which reduces scale effects and stabilizes variance. Each log-return series is centered and normalized to unit variance prior to analysis.

The 97 stocks are classified into 11 sectors using the Global Industry Classification Standard (GICS). To aid interpretation of within-sector and cross-sector dependencies, nodes are ordered by sector as shown in Table 7.1.

Table 7.1. Sector ordering of the 97 S&P 100 stocks used in the financial network analysis.

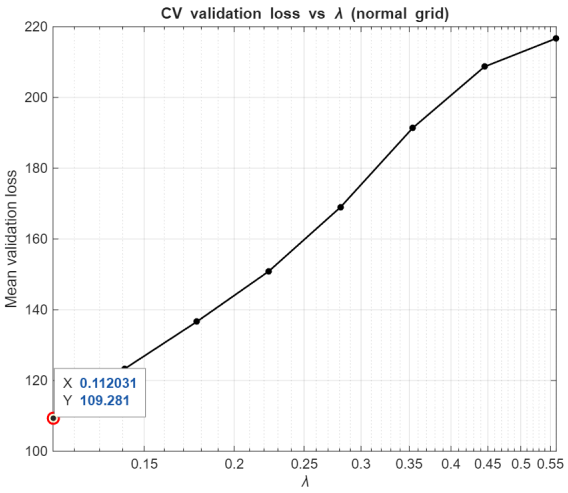
Sector	**Node indices**
Information Technology	1–12
Health Care	13–27
Financials	28–44
Real Estate	45–46
Consumer Discretionary	47–56
Industrials	57–68
Communication Services	69–76
Consumer Staples	77–87
Energy	88–92
Materials	93
Utilities	94–97

We estimate sparse multi-attribute networks and compare SS, BIC, and CV under the same framework as in Chapter 5, using both lasso and LSP.

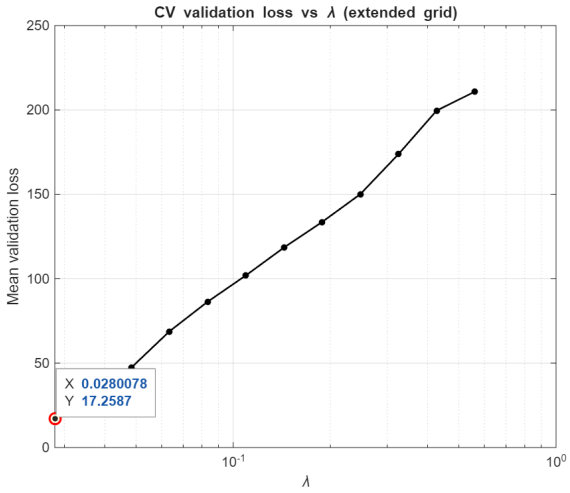
7.1 Estimated Financial Networks

We now describe the specific regularization grids and selected λ values for each method. For the lasso penalty, SS and CV initially used a grid in $[0.1120, 0.5602]$ (8 values), while BIC used a refined grid $[0.0280, 0.2801]$ (8 values). SS ($\lambda = 0.42224$) gives 135 edges (78 after pruning), BIC ($\lambda = 0.0751$) gives 1460 edges, and on this initial grid, CV ($\lambda = 0.11203$) gives 1107 edges. For LSP, SS uses a grid in $[0.0283, 0.1414]$ (8 values) with $\lambda = 0.040192$, resulting in 207 edges (124 after pruning). BIC uses a refined grid $[0.0071, 0.0707]$ (8 values), selecting $\lambda = 0.0098$ to obtain 686 edges. CV initially used the same grid as SS, selecting $\lambda = 0.028281$ and yielding 286 edges.

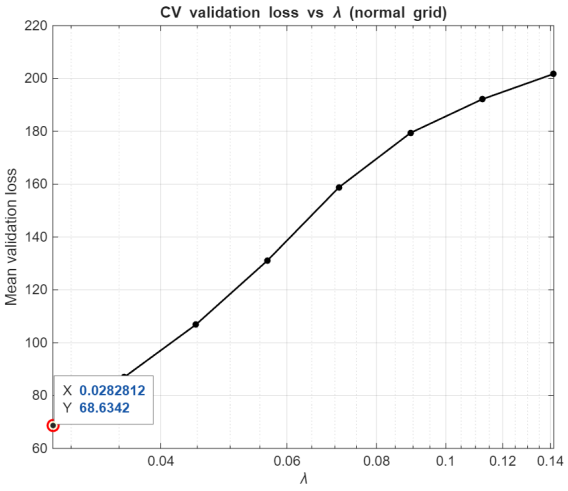
However, CV selected the lower boundary of its grid for both penalties, suggesting that the minimizer may lie below the tested range. To investigate, we extended the CV grids downward. For lasso, we expanded to $[0.0280, 0.5602]$. For LSP, we expanded to $[0.0071, 0.1414]$ with 12 values. On these extended grids, CV again selected the smallest available λ in both cases. For lasso, CV selected $\lambda = 0.028008$ and produced 2834 edges. For LSP, CV selected $\lambda = 0.00707$ and produced 995 edges. CV therefore favors smaller λ values and yields substantially denser graphs as the grid is extended. In contrast, SS and BIC produced stable, interpretable results within the same grids. This behavior is consistent with the simulation results in Chapter 6, where CV tends to overselect edges at moderate sample sizes while SS and BIC produce sparser estimates. Figure 7.2 displays the BIC score as a function of λ for both penalties, and Figure 7.3 shows the corresponding SS instability curves. In contrast to the monotonically decreasing CV error curves in Figure 7.1, both BIC and SS identify well-defined regularization levels: BIC through an interior minimum of the score, and SS through a threshold crossing of the instability measure. In all weighted adjacency plots, edge weights correspond to the groupwise Frobenius norms $\|\widehat{\boldsymbol{Z}}_{\lambda}^{(ij)}\|_F$ of the estimated precision matrix, except for the pruned SS graphs, where edges are selected by thresholding the stability weights $\widehat{\Pi}_{ij}(\lambda_{\mathrm{SS}})$ and the displayed weights are the corresponding group norms from the full-sample ADMM solution.



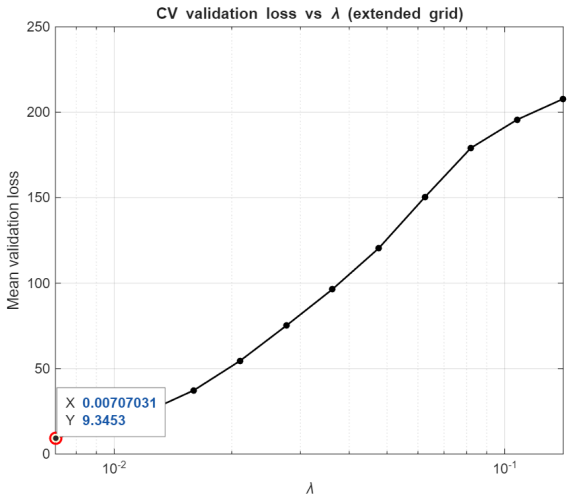
(a) Lasso with a normal grid. $\lambda = 0.11203$ at boundary.



(b) Lasso with an extended grid. $\lambda = 0.028008$ at new boundary.

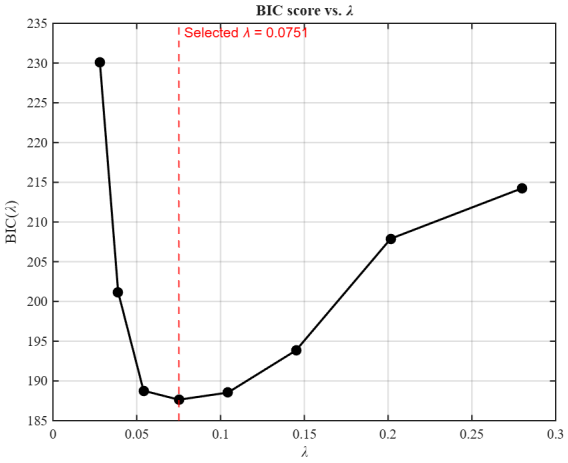


(c) LSP with a normal grid. $\lambda = 0.028281$ at boundary.

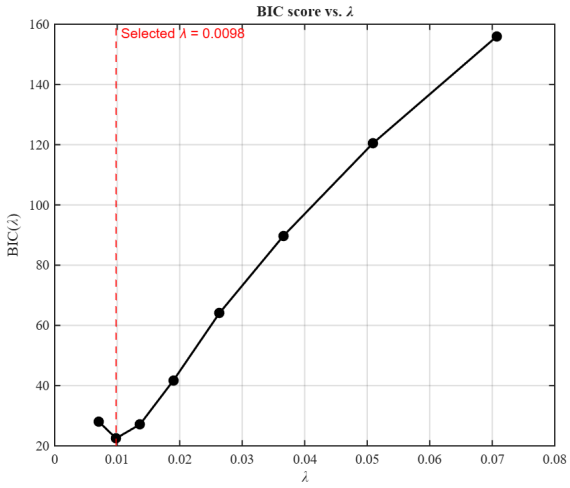


(d) LSP with an extended grid. $\lambda = 0.00707$ at new boundary.

Figure 7.1. CV error curves for normal and extended grids. For both penalties, CV selects the smallest λ on each grid. Extending the grid shifts the selection to a smaller value and yields denser estimated graphs.

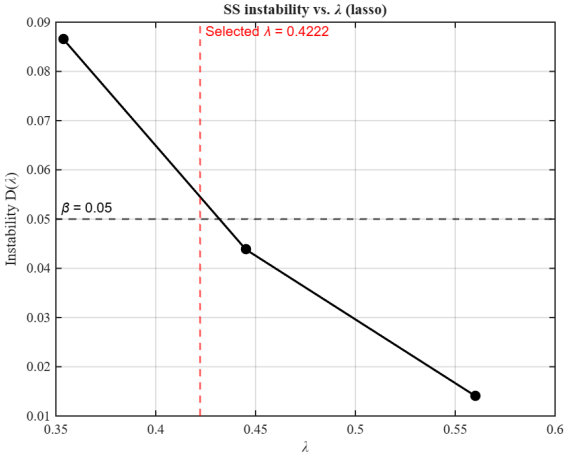


(a) BIC score with lasso. Minimum at $\lambda = 0.0751$.

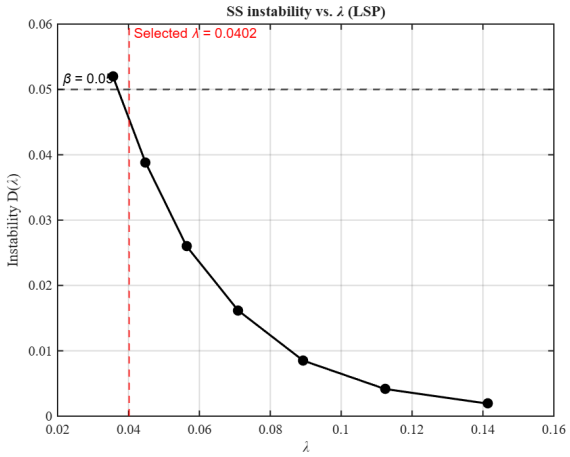


(b) BIC score with LSP. Minimum at $\lambda = 0.0098$.

Figure 7.2. BIC score curves for lasso and LSP on the S&P 100 dataset. Unlike the CV error curves in Figure 7.1, which decrease monotonically and select the boundary value, the BIC score exhibits a clear interior minimum for both penalties, indicating a well-defined trade-off between model fit and complexity.

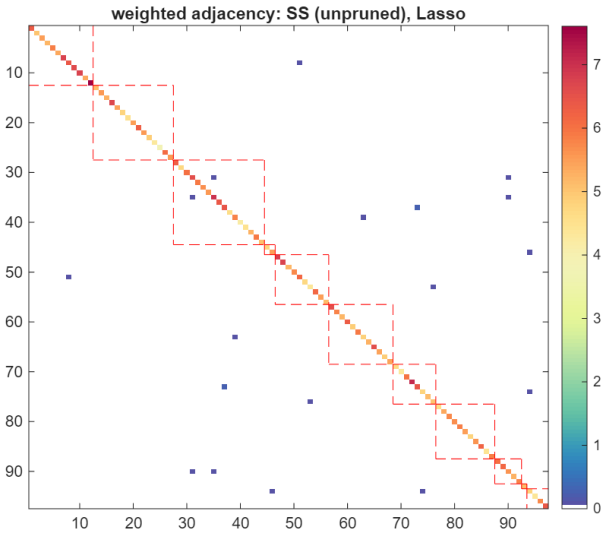


(a) SS instability with lasso. Selected $\lambda = 0.42224$.

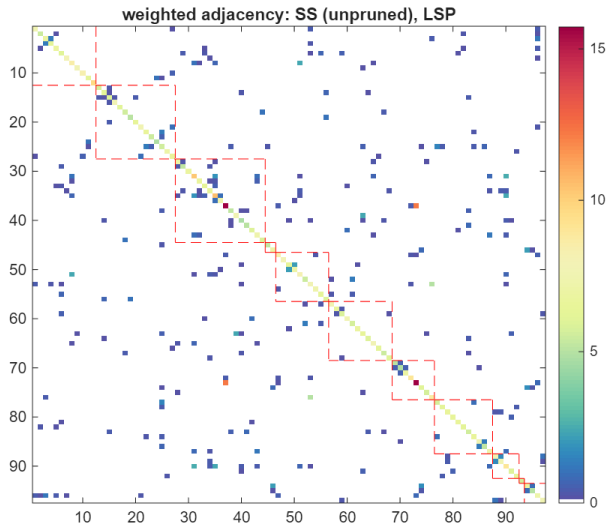


(b) SS instability with LSP. Selected $\lambda = 0.040192$.

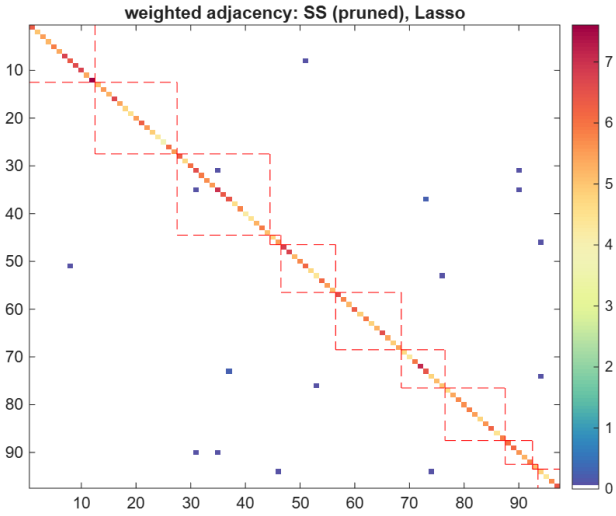
Figure 7.3. SS instability curves for lasso and LSP on the S&P 100 dataset. The dashed horizontal line marks the instability threshold β. SS selects the smallest λ whose monotonized instability does not exceed β, yielding a principled cutoff rather than the boundary selection observed with CV.



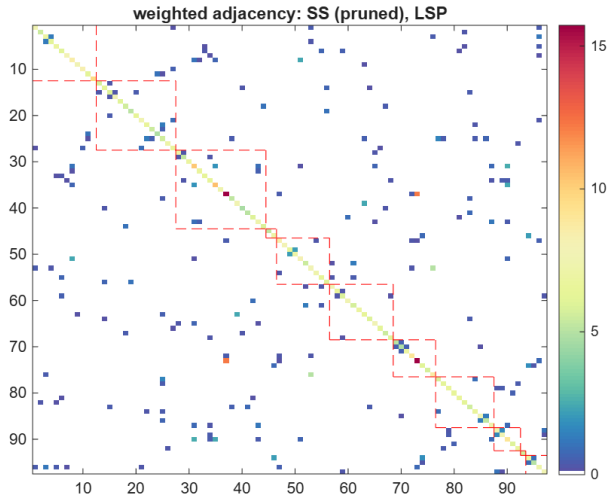
(a) SS with lasso (135 edges, unpruned)



(b) SS with LSP (207 edges, unpruned)

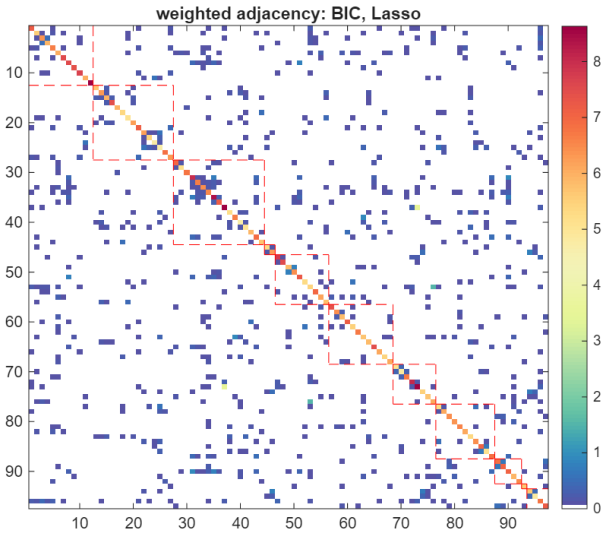


(c) SS with lasso (78 edges, pruned)

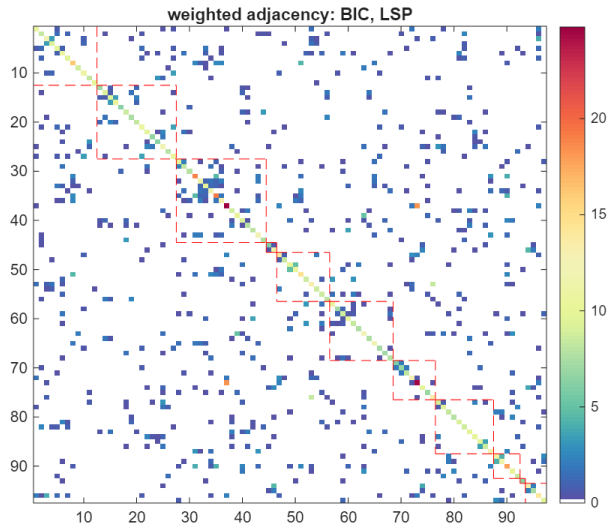


(d) SS with LSP (124 edges, pruned)

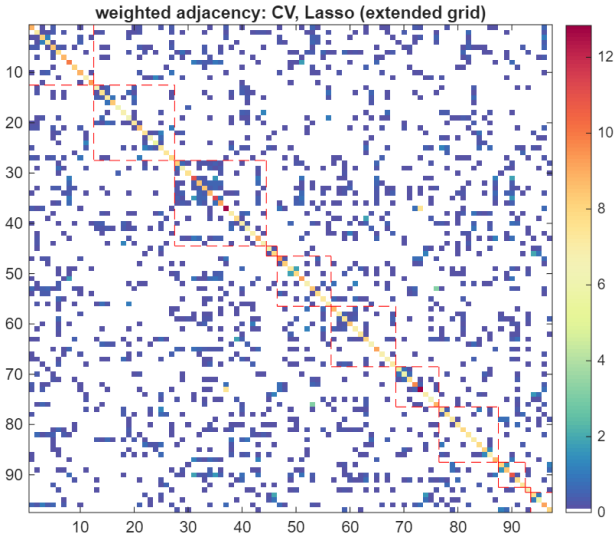
Figure 7.4. SS results for the S&P 100 financial network. Top row: unpruned graphs. Bottom row: pruned graphs ($p_{\text{thr}} = 0.60$). Left column: lasso penalty. Right column: LSP. Edge counts are shown in parentheses. Red dashed lines indicate sector boundaries.



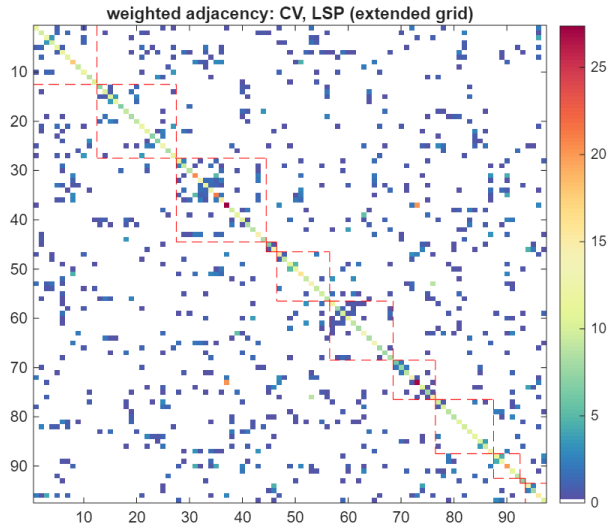
(a) BIC with lasso (1460 edges)



(b) BIC with LSP (686 edges)



(c) CV with lasso (2834 edges, extended grid)



(d) CV with LSP (995 edges, extended grid)

Figure 7.5. BIC and CV results for the S&P 100 financial network. Top row: BIC. Bottom row: CV with extended grid results. Left column: lasso penalty. Right column: LSP. Edge counts are shown in parentheses. Red dashed lines indicate sector boundaries.

7.2 Interpretation of Estimated Networks

The weighted adjacency matrices in Figures 7.4 and 7.5 reveal several structural patterns. We discuss within-sector clustering, cross-sector dependencies, and qualitative differences across selectors.

Within-sector clustering: Across all methods that produce interpretable graphs (SS and BIC under both penalties), the most prominent within-sector cluster corresponds to the financials sector (nodes 28–44). This block exhibits a dense concentration of edges with relatively high group-norm weights, consistent with the well-documented co-movement among banks, insurers, and diversified financial firms during the 2015–2020 period. The information technology sector (nodes 1–12) also shows notable within-sector connectivity, particularly under BIC with LSP and SS with LSP, reflecting shared exposure to technology-sector factors. Smaller but visible within-sector clusters appear for consumer staples (nodes 77–87), energy (nodes 88–92), and utilities (nodes 94–97). The communication services sector (nodes 69–76) shows a modest diagonal cluster under BIC and SS with LSP, though it is less pronounced than the financials or technology clusters.

Cross-sector dependencies: Several off-diagonal blocks indicate conditional dependencies between sectors. The most consistent cross-sector connections appear between information technology and financials (upper-left to financials block), and between financials and industrials (nodes 57–68). These cross-sector edges are visible under both BIC with LSP and SS with LSP (unpruned and pruned), suggesting that they reflect robust conditional dependencies rather than estimation artifacts. Under BIC with lasso, additional cross-sector edges appear more diffusely throughout the matrix, which may reflect the lasso's tendency to retain many small but nonzero entries compared to LSP.

Effect of selector and penalty on network structure: The choice of selector and penalty produces qualitatively different network estimates. SS with lasso (pruned) yields the sparsest graph (78 edges), retaining primarily within-sector edges along the diagonal with very few cross-sector connections; this represents the most conservative estimate. Note that in the SS lasso graphs, most off-diagonal edges have small group-norm weights and appear faint relative to the diagonal blocks. The SS lasso graphs appear visually sparser than the BIC graphs primarily because they contain far fewer edges, not because the individual edge weights are systematically smaller. SS with LSP (pruned, 124 edges) preserves the key within-sector clusters while revealing additional cross-sector structure, particularly between financials and other sectors. BIC with LSP (686 edges) produces an intermediate-density network in which both within-sector and cross-sector dependencies are clearly visible, and the sector-block structure remains well-defined. BIC with lasso (1460 edges) is denser, and while the financials cluster remains prominent, the off-diagonal regions become more populated, making sector boundaries harder to distinguish.

The CV estimates on extended grids produce the densest networks. CV with lasso (2834 edges) fills the off-diagonal regions nearly uniformly, obscuring the sector structure that is clearly visible under SS and BIC. CV with LSP (995 edges) is notably sparser than CV with lasso and produces a pattern that closely resembles BIC with LSP, suggesting that at this sample size ($N = 1258$) with the LSP penalty, CV begins to converge toward the BIC solution. This is consistent with the simulation results showing that CV with LSP becomes competitive at larger sample sizes. Since ground truth is unavailable for this dataset, these comparisons are necessarily qualitative. Nevertheless, the consistency between the real-data and simulation findings lends support to the practical recommendations developed in Chapter 8. We note, however, that the synthetic experiments assume i.i.d. observations, whereas financial time series typically exhibit temporal dependence. The qualitative agreement between the two settings is therefore encouraging but should be interpreted with this distinction in mind.

7.3 Summary

On the S&P 100 financial dataset, the three selectors produce qualitatively different network estimates. SS yields the sparsest graphs under both penalties (78 edges for lasso pruned, 124 for LSP pruned), identifying only the most reproducible conditional dependencies. BIC produces intermediate density (1460 edges for lasso, 686 for LSP), reflecting its balance between fit and complexity. CV, when given an extended grid, selects the densest graphs for both penalties (2834 for lasso, 995 for LSP) and consistently chooses the smallest available λ, confirming its tendency to prioritize predictive fit over parsimony. This relative ordering of selector sparsity is consistent with the patterns observed in the synthetic experiments of Chapter 6, where SS tends to be conservative, BIC balances likelihood and complexity, and CV favors denser graphs at moderate sample sizes. A detailed interpretation of these findings, along with practical recommendations, is provided in Chapter 8. We note, however, that the synthetic results assume i.i.d. data, whereas financial returns may exhibit temporal dependence, so the agreement between the two settings should be interpreted qualitatively.

Chapter 8

Discussion and Conclusions

This chapter interprets the main empirical findings reported in Chapters 6 and 7, and summarizes conclusions on the behavior of SS, BIC, and CV for MA-GGMs estimated using the lasso and the log-sum penalty. The discussion draws on the MA-GGM framework in Chapter 2, the penalized likelihood formulation in Chapter 3, and the model selection procedures in Chapter 4, all following [21].

8.1 Discussion of Main Findings

Effect of sample size: Sample size is the main driver of the relative performance of SS, BIC, and CV. For $N \in \{200, 400, 800\}$, CV performs poorly across BA, ER, and chain graphs under both penalties (Tables 6.1–6.14). For example, on BA graphs with $M_0 = 2$ and $N = 200$, CV with lasso yields F_1-scores below 0.10 and Hamming distances in the thousands, while SS with LSP and BIC with LSP recover significantly better edge sets (Table 6.1). A similar gap appears on sparse ER graphs with $\mathrm{fracp} = 0.01$ at $N = 400$ (Tables 6.12 and 6.14). At larger sample sizes, CV improves sharply when validation folds become large enough to stabilize the validation loss. In particular, CV with LSP achieves near-perfect recovery for $N \geq 2400$ in our settings (Tables 6.2 and 6.4). The same transition is visible on chain graphs (Tables 6.5 and 6.6), which suggests that the issue is not graph complexity but limited validation information.

BIC (robustness and computational efficiency): Across the simulation settings, BIC shows stable behavior and strong performance once N is moderate. When combined with LSP, BIC achieves high F_1-scores and low Hamming distances on both BA and ER graphs (Tables 6.1–6.14). For sparse ER graphs with $\mathrm{fracp} = 0.01$, BIC with LSP achieves near-perfect recovery by $N = 800$ (Tables 6.12 and 6.14). From a computational standpoint, BIC requires one fit per λ on the grid, while SS and CV require repeated refitting via subsampling or folds. Overall, BIC with LSP provides a strong balance between accuracy and cost, consistent with [21].

SS (strong small-sample performance): SS has a complementary profile to BIC in smaller sample regimes. On sparse graphs and for $N \in \{200, 400, 800\}$, SS with LSP often yields conservative edge sets with improved control of false positives (Tables 6.17–6.18). On ER graphs with $\mathrm{fracp} = 0.01$ at $N = 200$, SS with LSP (unpruned) outperforms BIC with lasso and all CV combinations (Tables 6.18 and 6.14). For BA graphs with $M_0 = 2$ at $N = 800$, SS with LSP (pruned) is competitive with BIC with LSP (Table 6.17). The pruning step can improve precision, while the instability threshold β provides direct control over sparsity. In Tables 6.17–6.18, we consider $\beta \in \{0.05, 0.075, 0.10\}$. Using the instability definition following [15], $\beta = 0.05$ emerges as a safe conservative default, though $\beta = 0.10$ with pruning can yield better Hamming distances when N is sufficiently large (see Chapter 6).

CV (limitations for structural recovery): The behavior of CV is consistent with its objective, which targets predictive negative log-likelihood rather than edge recovery. CV has been shown to overfit in high-dimensional regression settings [22], and its poor support recovery is also evident in our results. In our synthetic experiments, different λ values can yield similar predictive fit while producing very different graph structures, which explains the poor F_1-scores and large Hamming distances at $N \in \{200, 400, 800\}$ (Tables 6.1–6.14). The financial data further illustrates this: CV consistently selected the smallest λ on its grid for both penalties, and when the grid was expanded, it moved to the new boundary, producing progressively denser graphs (lasso: 1107 $\rightarrow$

2834 edges; LSP: 286 $\rightarrow$ 995 edges). When N is large, fold-level variability decreases and CV becomes competitive, especially with LSP (Tables 6.2 and 6.4).

Impact of graph topology and density: Topology and density affect all selectors. As density increases (larger fracp in ER or larger M_0 in BA), F_1 decreases and Hamming distance increases, reflecting a harder recovery problem with more edges and more opportunities for false discoveries. At comparable sparsity levels, BA graphs are harder to recover than ER graphs, which is consistent with the difficulty introduced by hub nodes and heavy-tailed degree distributions. Chain graphs are the easiest in our experiments, since their structured topology reduces ambiguity in edge selection.

Effect of penalty choice: Across nearly all settings, LSP improves structural recovery relative to lasso. For a fixed selector, LSP typically yields higher F_1-scores and lower Hamming distances, consistent with reduced shrinkage bias for non-convex penalties [4, 21]. For example, on BA graphs at $N = 800$, BIC with LSP achieves $F_1 = 0.97$ compared to 0.92 with lasso (Table 6.1). On ER graphs at $N = 200$, SS with LSP (unpruned) achieves $F_1 = 0.88$ versus 0.83 with lasso (Table 6.3). When computational resources permit, LSP is preferable in the regimes studied here. However, the advantage of LSP is not universal. At higher graph densities with limited sample sizes, lasso occasionally matches or slightly outperforms LSP. For example, on BA graphs with $M_0 = 5$ at $N = 400$, BIC with lasso achieves a higher F_1-score than BIC with LSP (Table 6.7), and on ER graphs with $\mathrm{fracp} = 0.10$ at $N = 400$, BIC with lasso yields a lower Hamming distance than BIC with LSP (Tables 6.13 and 6.14). In these denser settings with limited data, the convexity of the lasso may provide greater optimization stability compared to the non-convex LSP.

Connection to real-world financial data: The behavior observed on synthetic graphs is broadly reflected in the financial time series experiment in Chapter 7. SS produces the sparsest networks under both penalties (78 edges for lasso pruned, 124 for LSP pruned). BIC selects an intermediate number of edges (1460 for lasso, 686 for LSP). CV, when given an extended grid, produces

the densest graphs (2834 for lasso, 995 for LSP). Although ground truth is unknown in the real-data setting, this relative sparsity ordering mirrors the synthetic results at moderate sample sizes, supporting the practical conclusions drawn from the simulation study.

8.2 General Conclusions

The simulation study, together with the real-world financial data analysis, supports four main conclusions. First, CV is unreliable for structural recovery at small and moderate sample sizes and becomes competitive only when validation folds are sufficiently large (approximately $N \geq 1200$ with LSP in our settings). Second, BIC with LSP provides a robust trade-off between accuracy and computational efficiency across a wide range of settings. Third, SS with LSP is effective for sparse graphs and limited sample sizes, producing conservative and reproducible edge sets. Finally, no single criterion dominates in all regimes; the best choice depends on sample size, graph structure, and the study objective.

8.3 Practical Recommendations

Based on the empirical results from both synthetic experiments and real-world financial data, BIC with LSP is a strong default choice for structural inference in MA-GGMs when sample sizes are moderate or large. In sparse or small-sample settings where control of false positives and reproducibility are priorities, SS with LSP is recommended. In contrast, CV is best used when predictive performance is the primary objective and the sample size is sufficiently large to stabilize validation losses. For SS, we recommend $\beta = 0.05$ as a conservative default, with $\beta \in \{0.05, 0.075, 0.10\}$ providing a useful sensitivity range. These recommendations are supported by the consistent behavior observed across both controlled simulations and real-world financial network estimation.

8.4 Limitations

This study combines controlled simulations with a single real-world financial dataset, and several limitations remain. In the simulation experiments, only three graph families are considered and the dimension is fixed at $p = 100$, $m = 4$. Alternative graph structures, higher dimensions, or departures from Gaussian assumptions may yield different behavior. The real-data analysis is limited to one financial time series dataset and does not provide ground truth for structural recovery. As a result, comparisons on real data rely on relative sparsity and structural patterns rather than exact recovery metrics. We also focus on a specific class of penalties with fixed parameters ($\alpha = 0.05$, $\epsilon = 10^{-4}$, $p_{\text{thr}} = 0.60$), and different tuning choices may alter selector performance. Finally, the study is empirical in nature and does not establish new theoretical guarantees, which are addressed elsewhere in the literature [21].

8.5 Future Work

Future work includes extending the framework to differential multi-attribute Gaussian graphical models, where the objective is to identify changes in conditional dependence structure across multiple populations or experimental conditions.Additional directions include systematic evaluation across multiple real-world multi-attribute datasets from different domains, including but not limited to biological, environmental, and social network applications, to further assess robustness across various settings. Further study of alternative non-convex penalties and potentially more scalable optimization schemes will also be important for handling substantially larger dimensions. Finally, further investigating the theoretical conditions under which each selector achieves consistent model selection in the multi-attribute setting would greatly complement the empirical findings of this thesis.

Bibliography

[1] O. Banerjee, L. El Ghaoui, and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data," *Journal of Machine Learning Research*, vol. 9, pp. 485–516, 2008.

[2] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[4] E. J. Candès, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5, pp. 877–905, 2008.

[5] J. Chen and Z. Chen, "Extended Bayesian information criteria for model selection with large model spaces," *Biometrika*, vol. 95, no. 3, pp. 759–771, 2008.

[6] D. De Canditiis and S. Cirulli, "Model selection for inferring Gaussian graphical models," *Statistics and Computing*, vol. 34, no. 1, pp. 1–25, 2024.

[7] P. Erdős and A. Rényi, "On random graphs I," *Publicationes Mathematicae*, vol. 6, pp. 290–297, 1959.

[8] R. Foygel and M. Drton, "Extended Bayesian information criteria for Gaussian graphical models," in *Advances in Neural Information Processing Systems*, 2010.

[9] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation via the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[10] E. N. Gilbert, "Random graphs," *Annals of Mathematical Statistics*, vol. 30, no. 4, pp. 1141–1144, 1959.

[11] M. Kolar and E. P. Xing, "Consistent variable selection in sparse discriminative models," in *Advances in Neural Information Processing Systems*, 2010.

[12] M. Kolar, H. Liu, and E. P. Xing, "Markov network estimation from multi-attribute data," in *Proceedings of the 30th International Conference on Machine Learning*, 2013, pp. 55–63.

[13] M. Kolar, H. Liu, and E. P. Xing, "Graph estimation from multi-attribute data," *Journal of Machine Learning Research*, vol. 15, pp. 1713–1750, 2014.

[14] S. L. Lauritzen, *Graphical models*. Oxford University Press, 1996.

[15] H. Liu, K. Roeder, and L. Wasserman, "Stability approach to regularization selection (StARS) for high-dimensional graphical models," in *Advances in Neural Information Processing Systems*, 2010.

[16] Z. Lu, Y. Zhou, and J. Wu, "Efficient generation of scale-free networks via the Barabási–Albert model," *Physica A*, vol. 406, pp. 18–23, 2014.

[17] N. Meinshausen and P. Bühlmann, "High-dimensional graphs and variable selection with the lasso," *Annals of Statistics*, vol. 34, no. 3, pp. 1436–1462, 2006.

[18] N. Meinshausen and P. Bühlmann, "Stability selection," *Journal of the Royal Statistical Society: Series B*, vol. 72, no. 4, pp. 417–473, 2010.

[19] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *Journal of the Royal Statistical Society: Series B*, vol. 36, no. 2, pp. 111–147, 1974.

[20] J. K. Tugnait, "Sparse-group lasso for graph learning from multi-attribute data," *IEEE Transactions on Signal Processing*, vol. 69, pp. 1771–1786, 2021.

[21] J. K. Tugnait, "Multi-attribute graph estimation with sparse-group non-convex penalties," *IEEE Access*, May 2025, doi:10.1109/ACCESS.2025.3566959.

[22] L. Wasserman and K. Roeder, "High-dimensional variable selection," *Annals of Statistics*, vol. 37, no. 5A, pp. 2178–2201, 2009.

[23] Yahoo! Finance, "S&P 100 Index components — Historical data," `https://finance.yahoo.com`, Accessed: April 1, 2020.

Appendix A

Algorithmic Framework and Implementation Details

This appendix summarizes the optimization procedures and implementation details used throughout this thesis. The alternating direction method of multipliers (ADMM) solver and the local linear approximation (LLA) scheme for LSP follow the framework in [21]. We also provide algorithmic details for SS, BIC, and CV as implemented in the simulation study. Throughout this appendix, node-level quantities are defined over unordered node pairs $\{i, j\}$ with $i \neq j$, while scalar-level quantities used for element-wise penalties and BIC parameter counting are defined over scalar indices $(r, s) \in [mp] \times [mp]$. Symmetry of the precision matrix groups, $\mathbf{\Omega}^{(ij)} = \mathbf{\Omega}^{(ji)\top}$, implies that each node pair contributes once.

A.1 ADMM Optimization Framework

Let $\{\boldsymbol{x}(t)\}_{t=1}^{N}$ denote observations from the MA-GGM defined in Chapter 2. For a fixed regularization level $\lambda > 0$ and mixing parameter $\alpha \in (0, 1)$, the precision matrix is estimated by solving the sparse-group penalized likelihood problem introduced in Chapter 3. The precision matrix $\mathbf{\Omega} \in \mathbb{R}^{mp \times mp}$ is partitioned into $p \times p$ groups $\mathbf{\Omega}^{(ij)} \in \mathbb{R}^{m \times m}$. We denote the stacked dimension by $d = mp$. Penalties are applied only to off-diagonal groups, while diagonal groups are left unpenalized. The element-wise penalty $P_e(\mathbf{\Omega})$ and group penalty $P_g(\mathbf{\Omega})$, as well as the lasso and log-sum penalty functions, are defined in Section 3.2.

A.1.1 ADMM Formulation

To solve the sparse-group penalized Gaussian likelihood problem (3.6), we adopt an ADMM formulation following [3, 21]. ADMM is particularly well-suited here because it separates the optimization into simpler subproblems: the smooth Gaussian log-likelihood (which involves a log-determinant term) and the nonsmooth sparsity penalties can be handled in alternating steps. The key idea is to introduce an auxiliary variable $\boldsymbol{Z} \in \mathbb{R}^{mp \times mp}$ and impose the consensus constraint $\boldsymbol{\Omega} = \boldsymbol{Z}$. The variable $\boldsymbol{\Omega}$ handles the log-likelihood (which requires positive definiteness) while $\boldsymbol{Z}$ handles the sparsity penalties (which are separable). A scaled dual variable $\boldsymbol{U}$ enforces the constraint, and $\rho > 0$ is a penalty parameter that controls the strength of this enforcement. The resulting scaled augmented Lagrangian is

$$\bar{\mathcal{L}}_\rho(\boldsymbol{\Omega}, \boldsymbol{Z}, \boldsymbol{U}) = L(\boldsymbol{\Omega}) + \alpha P_e(\boldsymbol{Z}) + (1-\alpha) P_g(\boldsymbol{Z}) + \frac{\rho}{2}\|\boldsymbol{\Omega} - \boldsymbol{Z} + \boldsymbol{U}\|_F^2, \tag{A.1}$$

where $L(\boldsymbol{\Omega})$ is the Gaussian negative log-likelihood defined in (3.1). ADMM then proceeds by alternating minimization with respect to $\boldsymbol{\Omega}$ and $\boldsymbol{Z}$, followed by a dual variable update.

1) $\boldsymbol{\Omega}$ update (precision matrix step): Holding $\boldsymbol{Z}$ and $\boldsymbol{U}$ fixed, we solve for $\boldsymbol{\Omega}$:

$$\boldsymbol{\Omega}^{(k+1)} = \arg\min_{\boldsymbol{\Omega} \succ 0} \left[-\ln\det(\boldsymbol{\Omega}) + \operatorname{tr}\left(\widehat{\boldsymbol{\Sigma}}\,\boldsymbol{\Omega}\right) + \frac{\rho}{2}\|\boldsymbol{\Omega} - \boldsymbol{Z}^{(k)} + \boldsymbol{U}^{(k)}\|_F^2 \right]. \tag{A.2}$$

This subproblem combines the Gaussian log-likelihood with a quadratic penalty that keeps $\boldsymbol{\Omega}$ close to $\boldsymbol{Z}^{(k)} - \boldsymbol{U}^{(k)}$. Remarkably, it admits a closed-form solution via an eigenvalue decomposition. Define the symmetric matrix

$$\boldsymbol{A}^{(k)} = \widehat{\boldsymbol{\Sigma}} - \rho\big(\boldsymbol{Z}^{(k)} - \boldsymbol{U}^{(k)}\big) = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^\top, \tag{A.3}$$

where $\boldsymbol{P}$ is orthogonal and $\boldsymbol{D}$ is diagonal. The solution then applies a nonlinear shrinkage to each eigenvalue:

$$\boldsymbol{\Omega}^{(k+1)} = \boldsymbol{P}\boldsymbol{D}^{\star}\boldsymbol{P}^{\top}, \qquad D_{ii}^{\star} = \frac{-D_{ii} + \sqrt{D_{ii}^2 + 4\rho}}{2\rho}. \tag{A.4}$$

This formula resembles the quadratic formula because it arises from solving a first-order optimality condition that involves a quadratic term from the augmented Lagrangian. Geometrically, it shrinks the eigenvalues of $\boldsymbol{A}^{(k)}$ toward positive values, ensuring that $\boldsymbol{\Omega}^{(k+1)}$ remains symmetric positive definite, a crucial requirement for a valid precision matrix.

2) $\boldsymbol{Z}$ update (penalty step): With $\boldsymbol{\Omega}$ updated, we now solve for $\boldsymbol{Z}$. Define the intermediate matrix

$$\boldsymbol{Y}^{(k)} = \boldsymbol{\Omega}^{(k+1)} + \boldsymbol{U}^{(k)},$$

which represents the current target adjusted by the dual variable. The $\boldsymbol{Z}$ update then becomes a proximal operator for the sparse-group penalty, applied to $\boldsymbol{Y}^{(k)}$. First, element-wise soft thresholding handles the lasso penalty for a scalar x:

$$\mathcal{S}_{\tau}(x) = \operatorname{sign}(x)\max(|x| - \tau, 0). \tag{A.5}$$

This operator shrinks small entries toward zero, with the threshold τ controlling the strength. Secondly, group-wise shrinkage handles the group lasso penalty for an $m \times m$ matrix $\boldsymbol{B}$:

$$\mathcal{G}_{\tau}(\boldsymbol{B}) = \begin{cases} \left(1 - \dfrac{\tau}{\|\boldsymbol{B}\|_F}\right)\boldsymbol{B}, & \|\boldsymbol{B}\|_F > \tau, \\ \boldsymbol{0}, & \text{otherwise.} \end{cases} \tag{A.6}$$

This shrinks an entire group toward zero if its Frobenius norm falls below τ, otherwise it scales the group down proportionally. The thresholds are determined by the penalty parameters and the ADMM penalty ρ:

$$\tau_{e,rs} = \frac{[\boldsymbol{\Lambda}_1]_{rs}}{\rho}, \qquad \tau_{g,ij} = \frac{[\boldsymbol{\Lambda}_2]_{ij}}{\rho}, \tag{A.7}$$

where $[\mathbf{\Lambda}_1]_{rs} = \alpha\lambda$ for $r \neq s$ and $[\mathbf{\Lambda}_2]_{ij} = (1-\alpha)m\lambda$ for $i \neq j$. These thresholds are constant across off-diagonal indices. For each off-diagonal group (i, j) with $i \neq j$, we first apply soft-thresholding entrywise to $\boldsymbol{Y}_{ij}^{(k)}$, then apply Frobenius shrinkage to the resulting group:

$$[\boldsymbol{W}_{ij}^{(k)}]_{st} = \mathcal{S}_{\tau_e}\big([\boldsymbol{Y}_{ij}^{(k)}]_{st}\big), \qquad s, t \in [m], \qquad \boldsymbol{Z}_{ij}^{(k+1)} = \mathcal{G}_{\tau_{g,ij}}\big(\boldsymbol{W}_{ij}^{(k)}\big). \tag{A.8}$$

Symmetry is enforced by $\boldsymbol{Z}_{ji}^{(k+1)} = (\boldsymbol{Z}_{ij}^{(k+1)})^\top$, and diagonal groups are copied directly from $\boldsymbol{Y}^{(k)}$ without penalization.

3) Dual update: The scaled dual variable is updated to accumulate the residual between $\mathbf{\Omega}$ and $\boldsymbol{Z}$:

$$\boldsymbol{U}^{(k+1)} = \boldsymbol{U}^{(k)} + \mathbf{\Omega}^{(k+1)} - \boldsymbol{Z}^{(k+1)}. \tag{A.9}$$

This step drives the consensus constraint $\mathbf{\Omega} = \boldsymbol{Z}$ toward satisfaction over iterations.

Convergence is monitored using the primal residual

$$\boldsymbol{r}^{(k)} = \mathbf{\Omega}^{(k)} - \boldsymbol{Z}^{(k)}, \tag{A.10}$$

which measures how far the consensus constraint is from being satisfied, and the dual residual

$$\boldsymbol{s}^{(k)} = \rho\big(\boldsymbol{Z}^{(k)} - \boldsymbol{Z}^{(k-1)}\big), \tag{A.11}$$

which measures stability in the $\boldsymbol{Z}$ update. The stopping thresholds are set to

$$\varepsilon_{\text{pri}} = d\,\text{tol} + \text{tol}\,\max\{\|\mathbf{\Omega}^{(k)}\|_F, \|\boldsymbol{Z}^{(k)}\|_F\}, \qquad d = mp, \tag{A.12}$$

$$\varepsilon_{\text{dual}} = d\,\text{tol} + \text{tol}\,\frac{1}{\rho}\|\boldsymbol{U}^{(k)}\|_F, \tag{A.13}$$

with $\mathrm{tol} = 10^{-4}$. The algorithm terminates when both residuals fall below their thresholds, subject to a minimum of five iterations and a maximum of 200. To improve convergence, ρ is adapted based on the residual imbalance. If the primal residual dominates ($\|\boldsymbol{r}^{(k)}\|_F > 10\|\boldsymbol{s}^{(k)}\|_F$), the penalty ρ is doubled and $\boldsymbol{U}$ is halved to place more weight on satisfying the constraint. Conversely, if the dual residual dominates ($\|\boldsymbol{s}^{(k)}\|_F > 10\|\boldsymbol{r}^{(k)}\|_F$), ρ is halved and $\boldsymbol{U}$ is doubled to encourage stability in $\boldsymbol{Z}$. This adaptive scheme, common in ADMM implementations, balances primal and dual progress.

A.2 Local Linear Approximation for LSP

When the log-sum penalty (LSP) is employed, the sparse-group penalized likelihood problem becomes non-convex, making direct optimization challenging. We adopt the local linear approximation (LLA) strategy from [21], which replaces the non-convex penalty with a sequence of convex surrogates. At each outer iteration, the non-convex penalty is linearized around the current estimate, a weighted lasso problem is solved, and the process repeats.

We first solve the convex sparse-group lasso problem:

$$\widehat{\boldsymbol{\Omega}} = \arg\min_{\boldsymbol{\Omega}\succ 0} \left\{ L(\boldsymbol{\Omega}) + \alpha\lambda \sum_{\substack{r,s\in[mp]\\ r\neq s}} \big|[\boldsymbol{\Omega}]_{rs}\big| + (1-\alpha)m\lambda \sum_{\substack{i,j\in[p]\\ i\neq j}} \|\boldsymbol{\Omega}^{(ij)}\|_F \right\}, \tag{A.14}$$

using the ADMM algorithm described in Section A.1.1. This lasso solution provides a good initializer for the LLA iterations. Recall the log-sum penalty from (3.3):

$$\rho_\lambda(|u|) = \lambda\epsilon \ln\!\left(1 + \frac{|u|}{\epsilon}\right) \qquad \lambda > 0,\ \ 1 \gg \epsilon > 0.$$

Let u_0 denote the current estimate of a scalar parameter u. At each outer iteration, LLA replaces $\rho_\lambda(|u|)$ with its first-order Taylor expansion around $|u_0|$:

$$\rho_\lambda(|u|) \approx \rho_\lambda(|u_0|) + \rho'_\lambda(|u_0|)\big(|u| - |u_0|\big). \tag{A.15}$$

Since $\rho_\lambda(|u_0|)$ and $|u_0|$ are constant in the current iteration, the minimization effectively depends only on the term $\rho'_\lambda(|u_0|)\,|u|$:

$$\rho_\lambda(|u|) \;\approx\; \rho'_\lambda(|u_0|)\,|u|. \tag{A.16}$$

For the log-sum penalty, the derivative is $\rho'_\lambda(|u_0|) = \lambda\epsilon/(|u_0|+\epsilon)$, yielding adaptive weights of the form $c/(|u_0|+\epsilon)$. In our implementation, LLA is realized by replacing the fixed penalty matrices $\boldsymbol{\Lambda}_1$ and $\boldsymbol{\Lambda}_2$ with adaptive versions evaluated at the current ADMM splitting variable $\boldsymbol{Z}^{(t)}$. Since $[\boldsymbol{\Lambda}_1]_{rs} = \alpha\lambda$ for $r \neq s$ and $[\boldsymbol{\Lambda}_2]_{ij} = (1-\alpha)m\lambda$ for $i \neq j$, the LLA derivative directly yields the adaptive element-wise weights for scalar indices $(r, s) \in [mp] \times [mp]$ with $r \neq s$:

$$\lambda^{(t)}_{e,rs} = \frac{[\boldsymbol{\Lambda}_1]_{rs}}{\left|[\boldsymbol{Z}^{(t)}]_{rs}\right| + \epsilon}. \tag{A.17}$$

For node pairs $(i, j) \in [p] \times [p]$ with $i \neq j$, the adaptive group weights are

$$\lambda^{(t)}_{g,ij} = \frac{[\boldsymbol{\Lambda}_2]_{ij}}{\|\boldsymbol{Z}^{(ij)(t)}\|_F + \epsilon}. \tag{A.18}$$

Here, ϵ is the same parameter appearing in the log-sum penalty definition (3.3), set to $\epsilon = 10^{-4}$ throughout. With these adaptive weights fixed, each outer LLA iteration solves a convex weighted sparse-group lasso problem:

$$\widehat{\boldsymbol{\Omega}}^{(t+1)} = \arg\min_{\boldsymbol{\Omega}\succ 0} \left\{ L(\boldsymbol{\Omega}) + \alpha \sum_{\substack{r,s\in[mp]\\ r\neq s}} \lambda^{(t)}_{e,rs} \left|[\boldsymbol{\Omega}]_{rs}\right| + (1-\alpha)m \sum_{\substack{i,j\in[p]\\ i\neq j}} \lambda^{(t)}_{g,ij} \|\boldsymbol{\Omega}^{(ij)}\|_F \right\}. \tag{A.19}$$

Each subproblem is solved using the same ADMM routine as in the lasso case. In this thesis, the number of outer LLA updates is fixed to 1. A single reweighting step has been shown to capture most of the benefits of non-convex penalization [21], while keeping computational cost low.

A.3 Stability Selection

This section provides implementation details of SS that supplement the formulation in Section 4.2. For each dataset and each candidate $\lambda \in \Lambda$, SS uses $B = 20$ subsamples drawn uniformly without replacement from the N observations. Each subsample has size $b_N = \lfloor 10\sqrt{N} \rfloor$ [15]. For each subsample, we compute the sample covariance matrix and solve (3.6) to obtain the ADMM solution $(\widehat{\boldsymbol{\Omega}}_\lambda^{(b)}, \widehat{\boldsymbol{Z}}_\lambda^{(b)})$. Edge indicators, selection probabilities, instability, and monotone correction are computed as defined in Section 4.2 (equations (4.1)–(4.4)).

We select λ_{SS} as the smallest grid value whose corrected instability does not exceed $\beta = 0.05$. To refine λ beyond the grid, we run a bisection search between the two adjacent grid points whose corrected instabilities bracket β. The refinement runs for at most six iterations, or until $|\widehat{D}(\lambda) - \beta| \leq 0.01$. After selecting λ_{SS}, we report both the unpruned and pruned SS graphs as defined in Section 4.2. The unpruned graph uses the groupwise Frobenius norm rule on the full-sample ADMM solution at λ_{SS}. The pruned graph thresholds the stability weights $\widehat{\Pi}_{ij}(\lambda_{\mathrm{SS}})$ at $p_{\mathrm{thr}} = 0.60$ to form its own graph, as stated in the main text.

A.4 Bayesian Information Criterion

This section details the implementation steps of BIC that complement Section 4.3.

For each $\lambda \in \Lambda_{\mathrm{BIC}}$, we solve (3.6) to obtain the ADMM solution $(\widehat{\boldsymbol{\Omega}}_\lambda, \widehat{\boldsymbol{Z}}_\lambda)$. The BIC score is computed using (4.9), with the enlarged edge set defined in (4.8). The number of free off-diagonal parameters is $|\widehat{\bar{\mathcal{E}}}(\lambda)|/2$.

BIC is evaluated on a grid of $T = 8$ logarithmically spaced values. The grid construction follows Section 4.1: the BIC grid is defined by $\lambda_T^{\mathrm{BIC}} = \lambda_T/2$ and $\lambda_1^{\mathrm{BIC}} = \lambda_T^{\mathrm{BIC}}/10$, with intermediate values obtained by logarithmic spacing. The selected regularization parameter λ_{BIC} minimizes (4.9) over Λ_{BIC}, and is subsequently applied to the full dataset to obtain the final graph.

A.5 Cross-Validation

This section also provides additional details of the implementation of CV to complement Section 4.4. We use $K = 5$ folds.

The index set $\{1, \ldots, N\}$ is randomly partitioned into disjoint folds $\mathcal{I}_1, \ldots, \mathcal{I}_5$ of approximately equal size. For each fold f, the training set is $\mathcal{I}_{\text{tr}}^{(f)} = \{1, \ldots, N\} \backslash \mathcal{I}_f$ and the validation set is $\mathcal{I}_{\text{val}}^{(f)} = \mathcal{I}_f$. For each $\lambda \in \Lambda$, we fit the model on the training data by solving (3.6) to obtain $(\widehat{\boldsymbol{\Omega}}_\lambda^{(f)}, \widehat{\boldsymbol{Z}}_\lambda^{(f)})$. The validation loss is computed using (4.10). To stabilize the log-determinant evaluation, we replace $\widehat{\boldsymbol{\Omega}}_\lambda^{(f)}$ by $\widehat{\boldsymbol{\Omega}}_\lambda^{(f)} + \delta \boldsymbol{I}$ with $\delta = 10^{-6}$. The CV score averages (4.10) across the five folds, and λ_{CV} minimizes this average over the same grid Λ used for SS. The chosen regularization parameter is then applied to the full dataset to obtain the final CV graph.

Appendix B

Computational Tools and Environment

This appendix documents the computational environment and parameter settings used to produce all results in this thesis, ensuring that the experiments can be reproduced.

All simulations were performed in MATLAB R2024b on a Lenovo V15 G4 laptop running Windows 11 Pro with an AMD Ryzen 5 7530U processor. Algorithms were executed in serial without parallelization. A full experiment consisting of 50 Monte Carlo trials required between 21 and 30 hours, depending on sample size, graph density, and the processing power of the computational device.

B.1 Computational Reproducibility

- ADMM tolerances: $\tau_{\text{abs}} = \tau_{\text{rel}} = 10^{-4}$.
- Initial ADMM penalty: $\rho^{(0)} = 2.0$, with adaptive updates.
- Penalty mixing parameter: $\alpha = 0.05$.
- Log-sum tuning parameter: $\epsilon = 10^{-4}$.
- Number of LLA outer iterations: 1.
- Stability selection subsamples: $B = 20$.
- Stability selection subsample size: $b_N = \lfloor 10\sqrt{N} \rfloor$.

- Stability selection instability threshold: $\beta = 0.05$.
- SS refinement: bisection search with at most 6 iterations.
- SS pruning threshold: $p_{\text{thr}} = 0.60$.
- Cross-validation folds: $K = 5$.
- CV log-determinant regularization: $\delta = 10^{-6}$.
- Monte Carlo trials: 50 per configuration.
- Regularization grids: Λ for SS and CV (with $\lambda_1 = \lambda_T/5$), and Λ_{BIC} for BIC (with $\lambda_T^{\text{BIC}} = \lambda_T/2$, $\lambda_1^{\text{BIC}} = \lambda_T^{\text{BIC}}/10$), both with $T = 8$ logarithmically spaced values.
- Main MATLAB scripts, available at:

  `https://github.com/Worlasidzam/Multi-attribute-Gaussian-Graphical-Model-Toolbox`

  Synthetic experiments:

  `multiAttri_main_stab_allBic`

  `multiAttri_main_cv_only`

  Real-world data:

  `multiAttr_real_bic_all_new`

  `multiAttr_real_ss_all_new`

  `multiAttr_real_cv_all_new`

  Helper functions:

  `GenGraphPrec`

  `dataGen`

  `opt_admm_ma`

  `optimize_admm_ma`

  `opt_admm_ma_adap`

stab_selec_modv4

bic_selec

bisection_uplim

performance