8-15-2017

Information Theoretic Study of Gaussian Graphical Models and Their Applications

Ali Moharrer
Louisiana State University and Agricultural and Mechanical College, amohar2@lsu.edu

INFORMATION THEORETIC STUDY OF GAUSSIAN GRAPHICAL MODELS AND THEIR APPLICATIONS

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Electrical and Computer Engineering

by
Ali Moharrer
B.Sc., Zanjan University, 2010
M.Sc., Amirkabir University of Technology, 2013
December 2017

To my family, Shoulaj, Jafar, Hana, YasmanGoola, for their endless support
To my beloved wife, ZarPar

Acknowledgements

Firstly, I would like to express my sincerest gratitude to my advisor, Prof. Shuangqing Wei. I am grateful to him for his consistent and invaluable support and guidance that he provided me throughout my tenure as a PhD student at LSU. The beginning and completion of this long journey, full of challenges, would not have been possible without his generous support and efforts.

I want to express my special gratitude to all respected members of my advisory committee, Prof. Xin Li, Prof. Jian Zhang, Prof. Hongchao Zhang, and Prof. Jason Crow for their generous support and cooperation. Also, I would like to thank all my teachers of EECS for a great learning experience that I had during my graduate coursework at LSU.

I also wish to give my heartfelt thanks to my wife, ParPar Zafar, who has been a constant source of support and encouragement during the challenges of graduate school and life. I am truly thankful for having you in my life.

Also, my deep and sincere gratitude to my family for their continuous and unparalleled love, help and support. Indebted to my mother and my father for giving me the opportunities and experiences that have made me who I am. I always knew that you believed in me and wanted the best for me. You selflessly encouraged me to explore new directions in life and seek my own destiny. Thank you for teaching me that my job in life was to learn, to be happy, and to know and understand myself; only then could I know and understand others. This journey would not have been possible if not for you.

Finally, I am thankful to my dearest friends, Masoomeh-va-Akbar whose creative thoughts and intriguing conversations kept me motivated throughout my PhD years.

Table of Contents

Abstract

In many problems we are dealing with characterizing a behavior of a complex stochastic system or its response to a set of particular inputs. Such problems span over several topics such as machine learning, complex networks, e.g., social or communication networks; biology, etc. Probabilistic graphical models (PGMs) are powerful tools that offer a compact modeling of complex systems. They are designed to capture the random behavior, i.e., the joint distribution of the system to the best possible accuracy. Our goal is to study certain algebraic and topological properties of a special class of graphical models, known as Gaussian graphs.

First, we show that how Gaussian trees can be used to determine a particular complex system's random behavior, i.e., determining a security robustness of a public communication channel characterized by a Gaussian tree. We show that in such public channels the secrecy capacity of the legitimate users Alice and Bob, in the presence of a passive adversary Eve, is strongly dependent on the underlying structure of the channel. This is done by defining a relevant privacy metric to capture the secrecy capacity of a communication and studying topological and algebraic features of a given Gaussian tree to quantify its security robustness. Next, we examine on how one can effectively produce random samples from such Gaussian tree. The primary concern in synthesis problems is about efficiency in terms of the amount of random bits required for synthesis, as well as the modeling complexity of the given stochastic system through which the Gaussian vector is synthesized. This is done through an optimization problem to propose an efficient algorithm by which we can effectively generate such random vectors. We further generalize the optimization formulation from Gaussian trees to Gaussian vectors with arbitrary structures. This is done by introducing a new latent factor model obtained by solving a constrained minimum determinant factor analysis (CMDFA) problem. We discuss the benefits of factor models in machine learning applications and in particular 3D image reconstruction problems, where our newly proposed CMDFA problem may be beneficial.

Chapter 1
Introduction

1.1 Motivation

In many problems we are dealing with characterizing a behavior of a complex *uncertain* (non-deterministic) system or its response to a set of particular inputs. Such problems span over several topics such as almost all *machine learning* related problems; complex networks, e.g., social or communication networks; biology, etc. Because almost all the time we have only partial information about such complex systems, we can always assume that such systems have some latent and random factors. Since there are many of such complex systems, so defining a different framework for each of such complex systems is impractical. Therefore, one needs to define a common framework that can span through many of such problems and model such systems, to within certain accuracy.

Probabilistic graphical models (PGMs) are powerful tools that offer a compact and abstract modeling of complex *uncertain* systems [1]. They are designed to capture the random behavior, i.e., the *joint distribution* of the system to the best possible accuracy. In particular, PGMs are graph-based models of complex systems, and like graphs they consist of vertices and edges. The vertices in PGMs represent the *random variables* in a system. Such random variables capture different features and aspects of a system. The edges determine the inter-dependency between such random variables, and they can be either directed or undirected.

There are several reasons that make PGMs an effective tool in modeling complex systems. First, as we will see, by representing the joint distribution as a particular PGM, we exploit the dependency and more importantly *conditional independence* relations in a system. This way, in studying the system response one can easily ignore many unrelated features (random variables) and decrease the search space significantly. This is known as *factorizing* the joint distribution,

by considering the underlying *sparsity structure* of PGM, which makes the computations very efficient in *high-dimensional* problems where there are many features, and the PGM consists thousands of vertices and millions of edges. Second, when studying complex systems many times there are not enough informative features that can be extracted from the system. In such cases, we are dealing with *latent* models, in which many features are hidden to the observer and the goal is to infer the latent features only using observed and extracted features. Exploiting such sparsity in joint distribution is useful in understanding the semantic meanings of some of these hidden features. Finally, PGMs allow us to implement effective *inference* algorithms for estimating the objective functions in a system. Usually, such estimations are in a form of computing the *posterior probability* of a set of unknown features, given other known and observed features.

1.1.1 Gaussian Trees

As it can be seen PGMs can cover many fields and broad set of problems. Hence, in many situations we need to narrow down the system representation to a particular distribution and/or following certain sparsity patterns. Hence, in our research we focus on a special class of PGMs known as *Gaussian trees* [2]. The underlying structure of Gaussian trees is captured by *tree* graphs, i.e., connected undirected graphs with no cycles; and the joint distribution of the variables follows a Gaussian joint distribution. In particular, suppose $\mathbf{X} = \{X_1, X_2, ..., X_n\}$ defines the set of random variables in a model. Then, we assume $\mathbf{X}$ follows a joint Gaussian density with mean μ and a covariance matrix $\Sigma_{\mathbf{x}}$, and write $\mathbf{X} \sim N(\mu, \Sigma_{\mathbf{x}})$. We found the Gaussian density to be a well-known and widely studied distribution, while at the same time it can model many complex systems accurately.

As we will see in both Chapters 2 and 3 the Gaussian assumption leads to a convenient tractable results, due to extracted information from the *covariance matrix* $\Sigma_{\mathbf{x}}$. And the *zero* elements in the *precision matrix* (i.e., the inverse of a non-singular covariance matirx) correspond to conditionally independent features, i.e., those with no edges between them (non-adjacent variables).

1.2 Information Theory

Information Theory is a classical field of study in Communications that deals with topics such as data compression, coding, signal processing, etc. The most commonly used definitions in this topic are the notions of *entropy*, *mutual information*, and Kullback-Leibler (KL) divergence that each deal with characterizing the amount of information in a given data or determining the information theoretic distance between two models. While learning theory deals with algorithms to recover the underlying models and graphical structures, in information theory the goal is to determine fundamental limits and bounds (based on aforementioned metrics) under which the models can be estimated accurately. This is an important problem to address, since in many complex systems, we can only hope for proposing heuristics to address the problems, and information theory would be beneficial in characterizing the performance of such heuristics compared to ultimate achievable bounds.

Recently, there are several works that connect and adopt information theory metrics to learning problems, and in particular graphical models. The works in [3–5] introduce a new generalized KL distance metric to propose a recovery threshold for *community detection* problems. Such problem is addressed by relying on *channel coding theorems* in information theory. In particular, by looking at graphical models as a *codebook* consisting of particular graph parameters as *codewords*, we can translate a graphical model detection and estimation problems to a well-defined channel coding problem, where the fundamental limits to recover a transmitted codeword is completely characterized. The survey slides [6, 7] present several interesting information theoretic tools beneficial in community detection problems. Information theory can be helpful in graphical model selection and estimation problems, as the works [8, 9] propose information theoretic thresholds to lower bound the graphical model estimation error in certain settings.

Likewise, in our work we show (1) metrics such as mutual information are well-suited to address a well-defined security and privacy problem in graphical models; and (2) by optimizing the *model's randomness* which can be characterized by entropy metric, we can achieve efficient

encoding schemes and using an information theoretic distance similar to KL distance, we show the accuracy of the propose encoding method.

1.3 Summary of Research Work

We divide our study into two parts as follows:

1. Ordering Gaussian Trees based on Their Security Robustness Against Information Leakage.

2. Random Sampling from Latent Gaussian Trees.

We briefly discuss both these parts of our study in the subsequent sub-sections.

1.3.1 Ordering Gaussian Trees based on Their Security Robustness Against Information Leakage

In this section we show that how Gaussian trees can be used to determine a particular complex system's random behavior, i.e., determining a security robustness of a public communication channel characterized by a Gaussian tree. We show that in such public channels the secrecy capacity of the legitimate users *Alice* and *Bob*, in the presence of a passive adversary *Eve*, is strongly dependent to the underlying structure of the channel.

In particular, we rely on *topological* features and take the following steps to accomplish our goal:

1. Define a relevant privacy metric to capture the secrecy capacity of a communication

2. Define two game theoretic scenarios, i.e., min-max and max-min problems

3. Propose a new operation to transform one tree to another, with higher security robustness

4. Construct *posets* of Gaussian trees to order them based security performance

First, we define the conditional mutual information (CMI) $I(A; B|C)$, to capture the amount of leaked information from the Alice and Bob's channel exposed to Eve. In particular, CMI is shown to capture the number of secret key bits through public discussions between legitimate parties

Alice and Bob in the presence of a passive adversary eavesdropper Eve [10–12]. We assume all three agents have access to a set of n random variables whose joint probability density function (PDF) is featured in a graphical model, from which Alice and Bob choose two of the n variables, X_a and X_b, and Eve for another X_z, respectively. Then, we introduce the max-min and min-max scenarios, where either legitimate users or Eve get to choose first, respectively. We show that, given a particular Gaussian tree and depending on the scenario, which locations will be chosen by Alice, Bob, and Eve, to reach a compromise between information leakage and common information extractable.

Based on obtained conclusions we propose the *pruning and grafting* (PG) operation, by which one Gaussian tree can be transformed into a new structure with higher secrecy performance. We show that although the PG operation is local, however, it can impact both min-max and max-min values that are global metrics of a Gaussian tree.

Finally, based on PG operation we construct *partially order sets* (posets) of Gaussian trees that rank each tree based on its security performance. Each poset has a unique root known as *poset leader* which has the best secrecy performance among all the Gaussian trees in a poset.

We also study the *algebraic* properties of Gaussian trees and their role in determining the security robustness of such structures:

1. We assign a *Tutte-Like* polynomials to each Gaussian tree structure

2. We propose an algorithm based on *restricted integer partition* (RIP) method to enumerate the most secure structures

First, by studying a special class of polynomials, known as Tutte-like polynomials [13] we characterize each Gaussian tree with a particular polynomial. We prove that certain coefficients in such polynomials can characterize the security robustness of a tree relative to other structures in a poset.

Poset leaders are of great importance, since they are the most secure trees in each of their posets. We show that such structures have certain structural features, hence, we propose an efficient approach based on RIP methods to enumerate all such structures.

1.3.2 Random Sampling from Latent Gaussian Trees

In the problem that we have considered in Chapter 2 we assumed that the information of Gaussian tree, i.e., the source of randomness is explicitly and perfectly provided for all the users Alice, Bob, and Eve. Such assumption is no longer feasible in many situations where we deal with *hidden* (*latent*) variables. The hidden variables can be due to incomplete knowledge about the structure. Also, in Chapter 2 we assumed that the underlying structure follows Gaussian tree distribution. However, there are many situations where such constraint will be violated.

In Chapter 3 we relax both of the conditions by assuming that there are some hidden variables $\{Y_1, Y_2, ..., Y_k\}$ in the system that can influence the *observed* random vector $\mathbf{X}$. Also, the only information at hand are the samples from the random vector $\mathbf{X}$, i.e., the information of $\mu_{\mathbf{x}}$ and $\Sigma_{\mathbf{x}}$ is not given explicitly. Although we know that the underlying structure is still a tree, however, as we will see LGTs are general tree structures that contain Gaussian trees as a special case. We show that to obtain such hidden information we should rely on posterior Gaussian distributions of the form $p_{\mathbf{Y}|\mathbf{X}}$, i.e., given the observed variables what can we say about the hidden variables? More importantly, we propose an algorithm by which we can effectively generate such sample vectors $\mathbf{X}$. Then, the users based on the obtained samples can *estimate* the necessary statistics. Hence, the problem boils down to efficient synthesis of a random Gaussian vector with distribution $p_{\mathbf{X}}$. The primary concern in synthesis problems is about efficiency in terms of the amount of random bits required at the input, as well as the modeling complexity of the given stochastic system (i.e. channel) through which the Gaussian vector is synthesized. This is because *random number generators* (RNGs) are randomness resources, and algorithms should be designed in such a way to minimize the needed number of random bits for their corresponding tasks.

In several works [14–16] it is shown that the minimum amount of common randomness needed to approximate the joint density between a pair of random variables X and Y to be $C(X,Y) = \min_{X-W-Y} I(X,Y;W)$, where $C(X,Y)$ is widely known as *Wyner's common information*. Such quantity defines the necessary number of common random bits to generate correlated outputs X and Y. Similar to the above works, we also used the input code-rate to define the complexity of our encoding systems, since minimizing such rates results in reducing the number of common random bits needed to generate the output statistics. In particular, we propose a new *successive encoding* scheme that uses minimum number of random bits to generate a Gaussian random vector with joint prescribed joint distribution. We argue that such scheme uses the latent structure itself to generate samples that in fact makes the algorithm compact in terms of modeling complexity. Moreover, the accuracy of such method is shown by defining the *total variation* [16] distance between the synthesized density and the desired one.

We characterize the achievable rate region for the rate tuples of multi-layer latent Gaussian tree, through which the number of bits needed to simulate such Gaussian joint density are determined. The random sources used in our algorithm are the latent variables at the top layer of tree along with Bernoulli sign inputs, which capture the correlation signs between the variables. Given the derived achievable rate region for synthesis of latent Gaussian trees, we also quantify the amount of information loss due to unrecoverable sign information. It is shown that maximizing the achievable rate-region is equivalent to finding the worst case density for Bernoulli sign inputs where maximum amount of sign information is lost.

1.3.3 Common Information and Factor Models

In Chapter 3, we provide an operational approach to synthesize Gaussian vectors with underlying latent tree structure. However, in several cases we are dealing with complex systems that cannot be modeled accurately using Gaussian trees. Hence, in Chapter 4 we aim to generalize the optimization problem defined previously to any general Gaussian vector with arbitrary joint Gaussian distribution $p_{\mathbf{X}}$.

In particular, we formulate Wyner's common information for random vectors $\mathbf{x} \in \mathbb{R}^n$ with joint Gaussian density. We show that finding the common information of Gaussian vectors is equivalent to maximizing a log-determinant of the additive Gaussian noise covariance matrix. We coin such optimization problem as a *constrained minimum determinant factor analysis* (CMDFA) problem. We show the convexity of such problem, and then attempt on finding the necessary and sufficient conditions on CMDFA solution. We study the algebraic properties of CMDFA solution space. This is done through studying two extreme cases Gaussian graphical models, namely, *latent Gaussian stars*, and *explicit Gaussian chains*. Interestingly, we show that depending on pairwise covariance values in a Gaussian graphical structure, one may not always end up with the same parameters and structure for CMDFA solution as those found via graphical learning algorithms.

In Chapter 5 we show the benefits of such factor models in a specific class of learning problems, known as *image restoration and reconstruction*. We show the benefits of statistical methods developed in Chapter 4 and using Gaussian trees in accurate synthesis of 3D distorted images. Moreover, we will study the relation of our introduced optimization problems, which is founded on information theoretic metrics with several well-known optimization problems in learning field. Since we are dealing with high-dimensional data, such quantitative comparisons relies heavily on matrix operations borrowed from linear algebra.

Chapter 2

Extractable Common Randomness from Gaussian Trees: Topological and Algebraic Perspectives

In this chapter, we study both topological and algebraic properties of unrooted Gaussian trees in order to characterize their security performance. Such performance is measured by the corresponding potential in extracting common randomness from a given tree, which is further determined by max-min and min-max conditional mutual information values, subject to the order of selecting variables from the tree by legitimate nodes Alice and Bob, and an eavesdropper Eve, respectively. A new operation is proposed to transform a Gaussian tree into another, and also to order different Gaussian trees. Through such operation we construct several equivalent classes of Gaussian trees. Each class includes multiple Gaussian trees that can be partially ordered based on the associated max-min or min-max conditional mutual information (CMI) metric, and thus we can find the most secure and the least secure trees in each partially ordered set (poset). The union of all posets generates all possible non-isomorphic trees of the given number of variables. Then, we assign a particular polynomial to each Gaussian tree, and show that such polynomial can determine the relative security performance of the Gaussian tree with respect to other trees within the same class. In the end, based on a generalized integer partition method, we propose a novel approach to efficiently enumerate the most secure structures of all posets.

This study has resulted in three research papers, [17–19].

2.1 Introduction

Consider a problem of maximizing the number of established secret key bits through public discussions between two legitimate parties Alice and Bob in the presence of a passive adversary eavesdropper Eve [10–12]. In particular, assume three agents all have access to a set of n random variables whose joint probability density function (PDF) is featured in a graphical model, from which Alice and Bob choose two of the n variables, X_a and X_b, and Eve for another X_z,

respectively. It has been shown in [10–12] that the conditional mutual information (CMI) is the achievable secrecy rate through public discussion. It quantifies the number of secret key bits per channel use established between Alice and Bob, in the presence of passive Eve, who not only overhears all publicly exchanged messages, but also has a full access to X_z. Under this framework, the primary problem we are tackling in this paper is twofold: (1) Given a graphical model for the joint PDF of n variables, what variables should Alice and Bob select to maximize the amount of achievable secret information under Eve's attack? (2) How should we compare the potential secrecy level of different graphical models under properly defined metrics? Does there exist a consistent ordering of graphical models under which we could select the most favorable or least favorable models in terms of some properly defined metrics?

Such problems can find applications where secrecy shall be established between legitimate parties who need to decide what correlated random variables are to be chosen among a set of dependent candidates. For example, in a sensor network with n sensor-nodes whose measurements on physical parameters, say, temperature or humidity, are to be taken as sources of common randomness. Alice and Bob thus need to determine which two variables should be taken considering the leakage to a third party who can only compromise one of the remaining nodes. In addition, if there are options as to the set of sensor deployments in multiple set-ups, which result in multiple joint distributions of random variables, which topology of the underlying graphical model is more favorable? Even more interesting is what if we could transform the joint distribution under certain constraints by some local changes of sensor deployment, what guidance we could provide to such changes to attain a more favorable topology and joint distribution in terms of a larger amount of extractable secret key bits? Our goal is to provide insights and answers to these interesting and security related questions on graphical models.

As a first step, we have adopted a game theoretic framework to study the proposed problems: (1) max-min perspective: Alice and Bob first pick two random variables out of n variables, based on the pessimistic assumption that Eve will later choose the best variable from the remaining $n-2$

variables, i.e. to find the solution to the max-min conditional mutual information $I(X_a; X_b|X_z)$; (2) min-max perspective: Eve selects its favorite random variable first, while Alice and Bob choose from $n-1$ remaining random variables in the second place to find the solution to the min-max of the conditional mutual information. It should be noted that such a modeling has been coined as the security game in several contexts [20]. In fact, [20] define the secrecy capacity metric similar to the max-min value of conditional mutual information, which quantifies the maximum rate of reliable information transmitted from source to destination, in the presence of an eavesdropper. Due to the vast parameter space of graphical models, we restrict our attention to the cases where the joint PDF of n variables can be featured in unrooted Gaussian trees to address the aforementioned problems. Since in Gaussian trees there is a single path connecting any two variables, we found studying such models not only mathematically convenient, but also conducive to several fundamental insights. To solve the proposed problems, we have explored several results obtained in [21] regarding the conditional independence relationships in Gaussian trees.

As a constraint and also for the purpose of fair comparisons between Gaussian trees using either the max-min or min-max conditional mutual information, we require all weighted trees to share the same joint entropy, i.e., the same total amount of randomness. Consequently, we consider the secrecy levels of n random variables measured by either max-min or min-max CMI for cases where their joint distributions can be featured in un-rooted and weighted Gaussian trees under a constraint of a given total joint randomness, namely, the joint entropy.

The contributions can be categorized into two groups of topological and algebraic analysis:

- First, we show that in each scenario Alice, Bob, and Eve choose a triplet of nodes, where each node is in a special topological correspondence with others. As a result, our search space to find the max-min or min-max values will reduce significantly. Then, we formally define a pruning and grafting (PG) operation to transform one Gaussian tree into another. In particular, we introduce a special form of PG operation, namely, pruning and grafting of leaf with neighbor of degree 2 (PGLN-2). This operation has an important impact on

Gaussian trees: the max-min (min-max) value of the resulting Gaussian tree is always less than or equal to the max-min (min-max) value of the original Gaussian tree, hence making the resulting tree less secure than the original one. As a result, we form partially ordered sets (posets) of Gaussian trees and analyze the structural properties of these posets, by introducing an equivalent graph for each of these sets.

- Second, we introduce some algebraic tools to study the Gaussian trees and our introduced operations. We use the notion of Tutte-like polynomials [13] to represent each Gaussian tree with a specific polynomial. We show the impact of PGLN-2 operation on the corresponding polynomial, and also show that although non-isomorphic trees may have identical polynomials, however, for each of the posets and in the most cases this problem does not happen. We also show that some useful information can be extracted from this polynomial: the relative security of each Gaussian tree in a poset can be determined by its corresponding polynomial. Finally, we show that in each poset there exist the most secure Gaussian tree, as the poset leader, and introduce a systematic approach based on integer partitions [22] to effectively and directly enumerate all such structures.

2.1.1 Related Work

Other than our preliminary results in [23, 24] where partial findings were reported, to the best of our knowledge, there are no other previous works that fully capture the current problem.

The fundamental problem of secret key sharing between two users in the presence of an eavesdropper in a public discussion channel is considered in [10–12]. Manshaei et al. in [20] provide a game theoretic framework to introduce the secrecy capacity.

In [21, 25–27], some fundamental properties of Gaussian graphical models have been tackled using algebraic methods.

In [28], Patra and Lal propose an operation called *grafting* to order trees based on their *algebraic connectivity*, which is the second smallest eigenvalue (λ_2) of the Laplacian matrix. More-

over, in [29] the concept of *generalized tree shift* (GTS) is introduced to obtain certain posets for unlabeled and unweighted trees. It is further shown in [29] that the corresponding posets are also ordered based on the value of their algebraic connectivity.

In [23], we only considered the current problem for certain Gaussian trees with $n = \{4, 5\}$ nodes in the max-min scenario, where each random variable in a Gaussian tree has a unit variance. In [24], we extended the results of [23], to Gaussian trees with larger size. In this study, however, we generalize the scope into considering any Gaussian tree, with its random variables having arbitrary variance values. Also, we consider studying the min-max problem, as well as max-min scenario.

2.1.2 Organization

Section 2.2 presents the system model. We define the notion of Gaussian trees, and introduce the squared partial correlation coefficient to be used in the max-min and min-max scenarios. We study topological properties of Gaussian trees in Section 2.3. For both scenarios, we introduce the PG and PGLN-2 operations to order Gaussian trees. Furthermore, we formally define the equivalent classes of Gaussian trees, which are obtained by these operations. In Section 2.4 certain types of polynomials are introduced to characterize the security performance of each Gaussian tree. Also, we propose an effective method to enumerate certain Gaussian structures with robust security performances. Section 2.5 gives the concluding remarks.

2.2 System Model

Here, for the simplicity of denotations, instead of X_a, X_b, and X_z we use a, b, and z as the random variables that represent the choice of Alice, Bob, and Eve, respectively. The capital letters A, B, and Z denote the corresponding subsets of random variables chosen by each group.

In this study, we consider the Gaussian joint distribution to capture the density of n variables, *i.e.*, $P_{\boldsymbol{x}}(x_1, x_2, ..., x_n) \sim N(\mu, \Sigma)$, where μ is the mean vector that without loss of generality we assume $\mu = 0$, and Σ is a symmetric positive-definite covariance matrix of n random variables with $\sigma_{ij} \in \Sigma$ be the covariance between the random variables x_i and x_j. Furthermore, we assume that

the joint density can be characterized by a weighted and unrooted tree $T = (V, E, W)$, where V is the set of nodes representing the random variables, E is the set of edges showing the dependency relations between variables [25–27], and W is the set of edge weights with elements $w_{ij} = \sigma_{ij}$ whenever there is an edge between the nodes x_i and x_j. For a fair comparison between any two Gaussian trees, we assume that the users in all models have the same total amount of randomness, i.e., the same entropy. In this case, it is well known that the entropy of $\boldsymbol{x} = (x_1, x_2, ..., x_n)$ can be obtained by $H = 1/2 \ln((2\pi e)^n |\Sigma|)$ [30]. Hence, in order to obtain a fixed entropy we have to fix the determinant of the covariance matrix, i.e., $|\Sigma| = C_E$, where $C_E \in \mathbb{R}$ is a finite and non-zero constant. Suppose x_i, x_j, x_k are three variables drawn from the Gaussian graphical model, where $x_i \perp x_j | x_k$, i.e., x_i is conditionally independent of x_j, given x_k. Then, one can show that $\sigma_{ij} = \sigma_{ik}\sigma_{jk}/\sigma_{kk}$ [21], where σ_{kk} is the variance for the node x_k. Now, suppose there is a path $p_{ij} = e_{i,i_1} e_{i_1,i_2} ... e_{i_{m-1},j}$ between i and j consisting of m edges. Since in Gaussian trees there is only one path between any two vertices, we can write

$$\sigma_{ij} = \sigma_{i,i_1}\sigma_{i_1,i_2}...\sigma_{i_{m-1},j} / \prod_{l=i_1}^{i_{m-1}} \sigma_{ll} \tag{2.1}$$

Note that equation (2.1) is only valid for Gaussian tree models, hence makes studying these structures more convenient.

Next, we give a proper definition for the max-min problem, under the explained scenario.

Definition 1. Under the Gaussian tree model, legitimate entities Alice and Bob pick two nodes a and b on the tree under the attack by an eavesdropper Eve who selects the third and distinct node/variable z on the same tree. The objective of Alice and Bob is to select the pair (a, b) to maximize the minimum conditional mutual information $I(a; b|z)$. In particular, we adopt $\max_{\{a,b\}} \min_z I(a; b|z, T)$ as the first metric to measure the privacy level of a given weighted Gaussian tree $T = (V, E, W)$.

Similarly, we can define the min-max problem under the same circumstances. In this case we adopt $\min_z \max_{\{a,b\}} I(a;b|z,T)$ as another metric to measure the privacy level of a given weighted Gaussian tree $T=(V,E,W)$.

For Gaussian random variables the conditional mutual information $I(a;b|z)$ can be directly related to the *squared partial correlation coefficient*, which is defined as below [21],

$$\begin{aligned}\rho^2_{ab|z} &= \frac{(\sigma_{ab}-\sigma_{az}\sigma_{zz}^{-1}\sigma_{bz})^2}{(\sigma_{aa}-\sigma_{az}\sigma_{zz}^{-1}\sigma_{az})(\sigma_{bb}-\sigma_{bz}\sigma_{zz}^{-1}\sigma_{bz})} \\ &= 1-e^{-2I(a;b|z)}\end{aligned} \tag{2.2}$$

where $\sigma_{ab}=E[(a-\mu_a)(b-\mu_b)]$, the (a,b)-th element of Σ, is the covariance value between variables a and b (with both of them having zero mean). From (2.2), we can see that the conditional mutual information is a monotone increasing function of the squared partial correlation coefficient. Hence, in the following, we use partial correlation coefficient instead of the conditional mutual information as the security and privacy metric. Hence, the corresponding max-min and min-max values for a given Gaussian tree $T=(V,E,W)$ can be restated as:

$$\begin{aligned}S_M(T,W) &= \max_{\{a,b\}} \min_z \rho^2_{(ab|z,T,W)} \\ S_m(T,W) &= \min_z \max_{\{a,b\}} \rho^2_{(ab|z,T,W)}\end{aligned} \tag{2.3}$$

which will be used to compare and order different trees.

2.3 Topological Properties of Gaussian Trees

2.3.1 Structural Properties of the Triplet (a,b,z) in Both Max-Min and Min-Max Problems

Generally, to determine the security performance of a given Gaussian graphical model, we should solve both min-max and max-min cases, over all possible triplets (a,b,z). For Gaussian trees, however, we show in Lemma 1 that this large search space shrinks to a very small space, in which the triplet (a,b,z) should have certain structural relationships.

Lemma 1. For any Gaussian tree $T=(V,E,W)$,

1. The max-min value $S_M(T, W)$ is chosen from those set of triplets in which a and b are adjacent, and z is adjacent to either a or b [23].

2. The min-max value $S_m(T, W)$ is chosen from those set of triplets in which the node z is any internal (non-leaf) node, while a and b pick two adjacent nodes from the remaining vertices.

Proof. See Appendix 2.6.1. □

As expected, using Lemma 1 we can narrow down the large number of possible choices for both max-min and min-max cases to small subsets. Using Lemma 1 we can further simplify (2.2) and deduce the following formula for the squared partial correlation coefficient,

$$\rho^2_{ab|z} = \frac{\sigma^2_{ab}\sigma^2_{bz} - \sigma^2_{ab}\sigma_{bb}\sigma_{zz}}{\sigma^2_{ab}\sigma^2_{bz} - \sigma_{aa}\sigma^2_{bb}\sigma_{zz}} \tag{2.4}$$

Note that in (2.4) we implicitly assumed b is on the path from a to z, hence if in any case a lies in between b and z, then a and b should be switched in (2.4).

2.3.2 Pruning and Grafting Edges in Gaussian Trees

One simple way to produce all trees of given order is to begin with any arbitrary structure and move one of its leaf edges to somewhere else, in order to obtain new structures. Note that this method may result in many *isomorphic* (redundant) tree structures, which should be eliminated from the list. In particular, consider the trees shown in Figure 2.1. Tree T_2 is obtained from T_1 by moving the edge e; we call this particular operation as *pruning and grafting* (PG) operation. In other words, we prune the edge e from the node n_1 and graft it to the node v', to obtain T_2. For Gaussian trees, we formally define the PG operation as follows:

Definition 2. Consider a Gaussian tree $T_1 = (V, E_1, W_1)$ shown in Figure 2.1. The pruning and grafting (PG) operation refers to cutting the leaf edge e from n_1 and attaching it to some other node, namely, v', to obtain the Gaussian tree $T_2 = (V, E_2, W_2)$. Note that W_1 is any arbitrary set of edge-weights, and W_2 is obtained from W_1 by changing the covariance values associated with

the altered edge. In particular, in PG operation we assume all the covariance values (including all the variances) in the covariance matrix remain unchanged, except those values that are related to the altered node, namely, n_2.

Note that since in PG operation we essentially only move the edge e to some other place, all other structures shown in Figure 2.1 (including everything in the clouds) remain unchanged. As a result, it is reasonable to assume that all the variances (including $\sigma_{n_2n_2}$) remain the same; also all the covariance elements except those values that are related to the altered edge remain unchanged. In particular, let us define $\sigma_{v_in_2}$ and $\sigma'_{v_in_2}$ as the covariance values between any node $v_i \in V\backslash n_2$ in T_1 and T_2, respectively. Then in general $\sigma_{v_in_2} \neq \sigma'_{v_in_2}$, for all $v_i \in V\backslash n_2$. The other elements in both covariance matrices corresponding to the Gaussian trees T_1 and T_2 are equal. In Lemma 2 we show how to compute these altered covariances for a new tree.

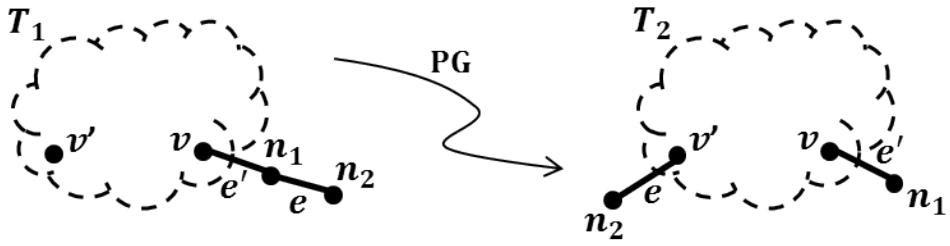


FIGURE 2.1: Pruning and grafting operation performed on edge e in tree T_1

Lemma 2. Consider any Gaussian tree $T = (V, E, W)$, with order $|V| = n$. We denote $|\Sigma_T|$ as the determinant of covariance matrix corresponding to T. Considering the PG operation shown in Figure 2.1, which transforms the Gaussian tree T_1 into T_2, with $|\Sigma_{T_1}| = |\Sigma_{T_2}|$. Let us denote $\sigma_{n_1n_2}$ and $\sigma'_{v'n_2}$ as the covariance values between the pairs (n_1, n_2) and (v', n_2) in trees T_1 and T_2, respectively; then we have $\sigma^2_{n_1n_2}/\sigma_{n_1n_1} = \sigma'^2_{v'n_2}/\sigma_{vv}$.

Proof. See Appendix 2.6.2. □

In the next sections, we discuss the importance of such results in determining the security performance of different Gaussian trees.

2.3.3 Pruning and Grafting Certain Leaf Edges in Gaussian Trees

In [28], Patra and Lal propose an operation called *grafting* to order trees based on their *algebraic connectivity*, which is the second smallest eigenvalue (λ_2) of the Laplacian matrix. Here, we introduce a new operation on Gaussian trees to obtain the ordering among different structures. We specify this operation as *pruning and grafting of leaf with neighbor of degree* 2 (PGLN-2). We show that by PGLN-2 one can change the security performance of Gaussian trees.

Definition 3. Consider a Gaussian tree $T_1 = (V, E_1, W_1)$ shown in Figure 2.2. The PGLN-2 operation refers to cutting the edge e and attaching it to the other end of its parent edge, i.e., e', to obtain the Gaussian tree $T_2 = (V, E_2, W_2)$. In fact, we can interpret PGLN-2, as a particular operation $\phi(.)$ acting on edge e in T_1, to produce the tree T_2, i.e., $\phi(T_1, e) = T_2$. Note that all the constraints regarding covariance values given in Definition 2 also hold in this case.

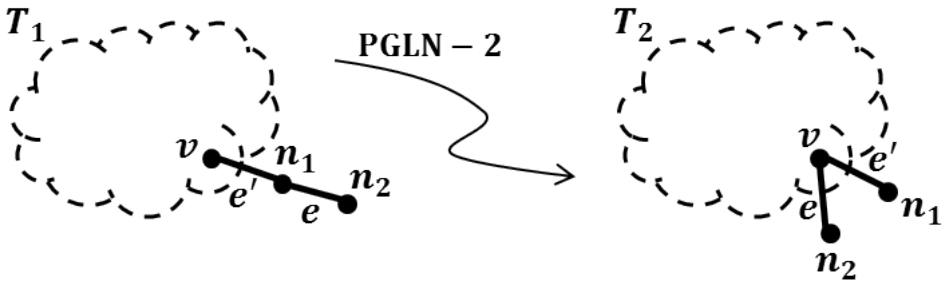


FIGURE 2.2: T_2 is obtained from T_1 by PGLN-2 operation

Note that $\phi(.)$ is not an *injective* mapping, since there might be two distinct edges (with their neighbors having degree of 2), say $e, e' \in E_1$ that $\phi(T_1, e)$ is isomorphic to $\phi(T_1, e')$. Also, by Lemma 2 we can conclude that $\sigma^2_{n_1n_2}/\sigma_{n_1n_1} = \sigma'^2_{vn_2}/\sigma_{vv}$. We have the following Lemma, whose proof can be found in Appendix 2.6.3.

Lemma 3. Consider the Gaussian trees shown in Figure 2.2, where $T_2 = \phi(T_1, e)$. Note that W_1 is any arbitrary set of edge-weights, and W_2 is obtained from W_1 (by changing the covariance values associated with the altered edge). Suppose the max-min values for T_1, and T_2 are $S_M(T_1, W_1)$ and $S_M(T_2, W_2)$, respectively. Also, $S_m(T_1, W_1)$ and $S_m(T_2, W_2)$ are the corre-

sponding min-max values for T_1 and T_2, respectively. We have $S_M(T_1, W_1) \geq S_M(T_2, W_2)$ and $S_m(T_1, W_1) \geq S_m(T_2, W_2)$.

Intuitively, for the max-min case using PGLN-2 operation on the edge e we are essentially adding another neighbor to the node n_2, hence giving more options to the eavesdropper to choose the best location to attack, resulting in smaller max-min values. On the other hand, although in the min-max case z cannot choose n_1 anymore (since it became a leaf), it can choose the node v (which has a higher degree now), thereby decreasing the number of possible choices for the pair (a, b). As we can see from Lemma 3, for any given Gaussian tree structure, the PGLN-2 operation always decreases both max-min and min-max values of the resulting Gaussian tree. As a result, this specific operation generates a certain ordering of Gaussian trees, in which the corresponding structures are ordered with regard to their respective max-min and min-max values. In the following, we formally define the tree ordering using the results obtained in Lemma 3,

Definition 4. Consider the trees $T_1 = (V, E_1, W_1)$ and $T_2 = (V, E_2, W_2)$, where $T_2 = \phi(T_1, e)$, for some leaf edge $e \in E_1$ that has a neighbor with degree two. We know from Lemma 3 that $S_M(T_1, W_1) \geq S_M(T_2, W_2)$ and $S_m(T_1, W_1) \geq S_m(T_2, W_2)$. In this setting, we write $T_1 \succeq T_2$, where the binary relation " $\succeq$ " shows the ordering of these trees, i.e., T_1 is more secure than T_2.

As we will see shortly, the ordering defined in Definition 4 leads to an interesting concept: we can define several *classes* for all Gaussian trees, and each class is a particular poset of distinct Gaussian trees.

While from Lemma 3 one may anticipate that any general PG operation results in less secure Gaussian trees, in the following proposition whose proof can be found in [31], we show that this is not the case.

Proposition 1. Consider the trees shown in Figure 2.1. Given a Gaussian tree $T_1 = (V, E_1, W_1)$, if we perform PG operation on the edge e to obtain the Gaussian tree $T_2 = (V, E_2, W_2)$, where $v' \neq v$. Then, in general $T_1 \not\succeq T_2$, i.e., T_1 is not always more favorable than T_2.

From Proposition 1 we can see that if two trees are not related through one or more PGLN-2 operations, then in general they cannot be ordered using our defined binary relation. In fact, without assigning a specific covariance matrix to each Gaussian tree, these structures cannot be consistently compared. This motivates us to search for certain structures that cannot be compared with each other, and at the same time they cannot be improved further, using PGLN-2 operation. In particular, we form sets of Gaussian tree structures, where each set contains a unique leader that is the most favorable topology among all topologies in the same set. Other topologies in a poset might be comparable/incomparable with each other. By classifying the trees into certain sets we can further study both their topological and algebraic properties.

2.3.4 Forming the Posets of Gaussian Trees

Based on the obtained results in Proposition 1 and Lemma 3 we can form posets [32] of Gaussian trees. Each poset is formed from its most favorable (MF) structure, $T_M = (V, E_M, W_M)$. In other words, T_M is the poset leader acting as the *ancestor* to all other Gaussian trees in the same poset, i.e., all other Gaussian trees can be obtained from T_M using one or more PGLN-2 operations (composition of several $\phi(.)$ functions). Also, in each poset given two trees T and T', they are adjacent if $T' = \phi(T, e)$, for the leaf edge $e \in E(T)$ that is connected to the a node having degree of two. Note that by Definitions 2 and 3, via the PGLN-2 operation only the covariance values regarding to the altered node will be changed, while the determinant of the covariance matrix corresponding to the trees remain the same. Hence, all trees in the same poset have a fixed determinant for their corresponding covariance matrices. Moreover, in Lemma 4, whose proof can be found in [31] we show the uniqueness of LF structures in each poset.

Lemma 4. In any poset with a given $T_M = (V, E_M, W_M)$ acting as a poset leader, we can find a unique least favorable (LF) structure, $T_L = (V, E_L, W_L)$, which acts as a *descendant* to all other trees.

Hence, we observe that our defined posets are certain classes of posets, which have a unique MF and LF structures. Also, from the results in Lemma 3 we know that T_M has the most secure structure, while T_L has the least secure structure in each poset. As an example, Figure 2.3 shows all six posets of Gaussian trees on 8 nodes. The posets consist of several unlabeled trees. Each poset consists of several Gaussian trees, and while such trees are weighted and consequently labeled, we consider them as being unlabeled. This is because by considering labeled trees, we are producing Gaussian trees (with isomorphic unlabeled structures) capturing different joint densities but with exactly the same security performance, making the obtained trees redundant. In Figure 2.3, the MF and LF structures are placed at the top and bottom of each poset, respectively. Note that in this figure, posets 1, 2, 3 and 6 are the special cases where posets are basically formed as fully ordered sets, hence any tree structure in each of these posets can be compared to other trees in the same poset, through one or more PGLN-2 operations. On the other hand, in each of the posets 4 and 5 there are some structures that cannot be compared using the rules given in Lemma 3. Note that beginning from any tree and performing several PGLN-2 (or performing reverse PGLN-2) we can obtain all other trees in the same poset. However, if we begin from MF structure, then by performing only PGLN-2 (and not its reverse) we can produce all other structures in the poset. In other words, the MF structures, acting as the poset leaders, can fully describe the poset structure. On the other hand, we also know that the MF topologies are the most secure trees in each poset. Hence, finding such structures is of huge importance, and there should be a method to systematically obtain these topologies. Thus, in section 2.4, we propose an efficient algebraic approach to enumerate all these structures systematically.

2.3.5 Directed Super-Graph Corresponding to Each Poset

From now on, for the simplicity of notations we call the leaf edges that are connected to a node with degree two as *special leaf edges*. Figure 2.3 gives us an intuition in order to construct a directed super-graph containing Gaussian trees. In particular, each poset can be converted into a directed super-graph $G = (V_s, E_s)$, where V_s is the set of trees in a poset acting as vertices, and

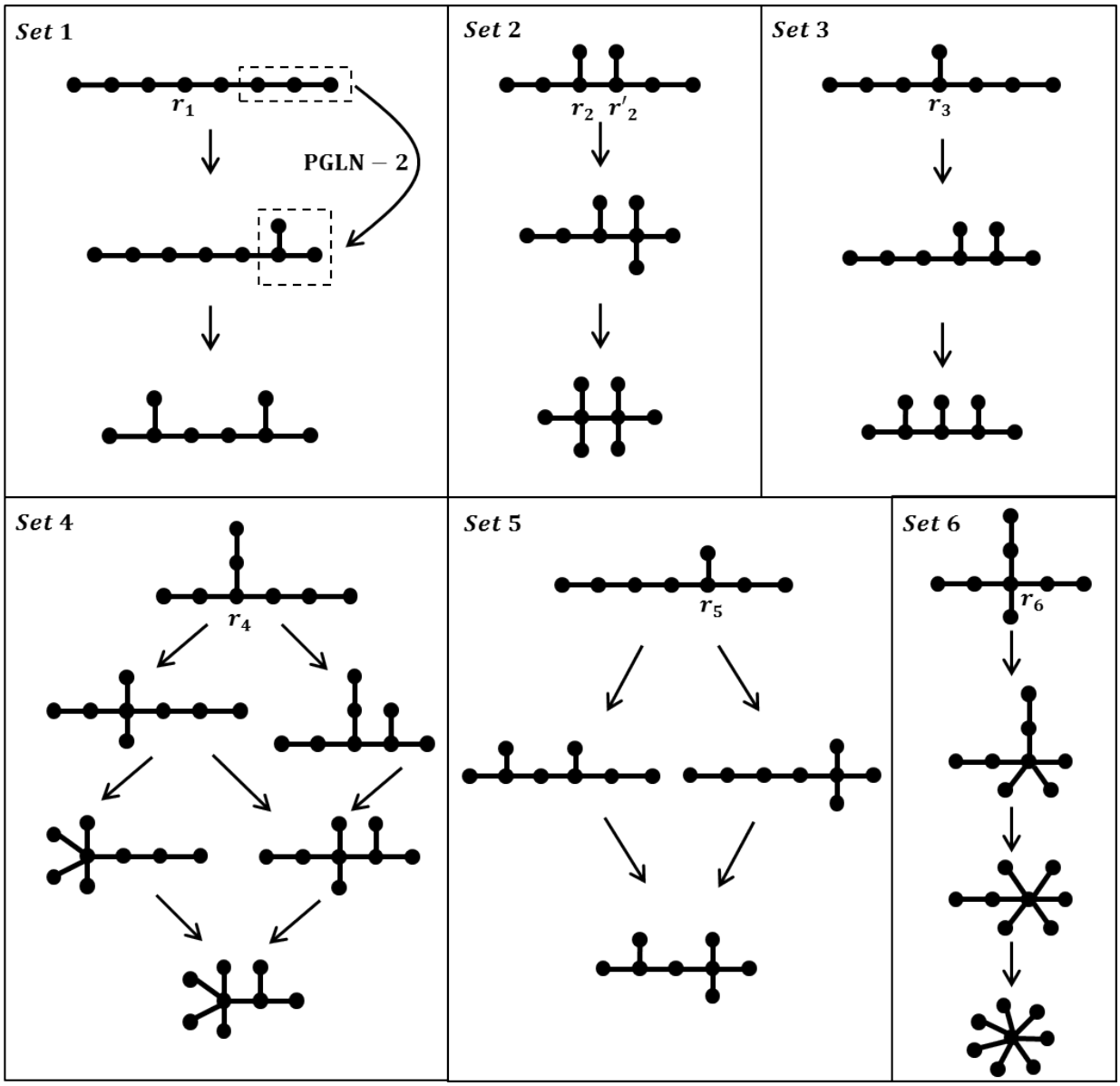


FIGURE 2.3: All the possible posets for Gaussian trees with $n = 8$ nodes

E_s is the set of directed edges between the two nodes that can be related using PGLN-2. Using this super-graph, we can easily identify the comparable tree structures: If there is a directed path between two structures, then they are comparable. Hence, we can conclude that both MF and LF structures can be compared with any other tree in a poset. For example, in Figure 2.3, in posets 1, 2, 3, and 6 there is a directed path between any tree structure so all the trees are comparable with each other, making each poset a fully ordered set. On the other hand, in posets 4 and 5

there are certain trees with no directed path between them, making such structures incomparable with each other under Lemma 3 conditions. In each poset, the MF structure has the maximum number of special leaf edges, while the LF structure has none of such edges. Also, observe that the poset leader fully characterizes the structure of its super-graph. In particular, the number of those special leaf edges in MF structure specifies the *length* (number of consecutive grafting operations plus 1) of the super-graph. Moreover, the structure of those special leaf edges specifies the *width* of the super-graph. For example, in Figure 2.3 we can see that the poset 2 has two special leaf edges, hence the super-graph has length 3. Also, since these special leaf edges are fully *symmetric* with respect to each other (performing PGLN-2 operation on either of those edges, results in an isomorphic tree structure), the poset 2 becomes fully ordered. On the other hand, in poset 5 because of the two *asymmetric* special leaf edges we obtain two different topologies in the next level. Roughly speaking, if those special edges become more symmetric, the poset tends to become fully ordered.

Although converting each poset to its corresponding super-graph simplifies the comparison between tree topologies in a set, as it can be observed, for larger trees identifying these special branches and ordering trees by grafting operation becomes more challenging. Hence, in the following, we aim to study the tree structures and their associated posets in a more abstract and general way.

2.4 Algebraic properties of Gaussian Trees

2.4.1 Tutte-Like Polynomials for Gaussian Trees in Posets

In this section, in order to model the Gaussian trees and the corresponding posets more systematically, we study the algebraic properties of these models. As we may see in the following, these properties will further help us characterize the special leaf edges, and thus allow us to evaluate the security robustness of any tree structure within a poset. To achieve this goal, for each tree, we associate a two-variable *Tutte-like* polynomial defined in [13], where Chaudhary and Gordon modify the definition of the Tutte polynomial to obtain a new invariant for both rooted and unrooted trees.

Also, they proved that this polynomial uniquely determines rooted trees. For unrooted trees however, it is shown in [33] that certain classes of caterpillars have the same polynomials assigned to them. However, interestingly, we prove that in each poset, in many cases the trees have unique polynomials.

Let $R(T)$ denote the collection of all subtrees of T, and $L_E(S)$ denote the leaf edges in the subtree S, i.e., the edges that are connected to leaf nodes, then we have [13],

$$f_E(T;t,z) = \sum_{S \subseteq R(T)} t^{|E(S)|}(z+1)^{|E(S)|-|L_E(S)|} \tag{2.5}$$

where $|E(S)|$ is the total number of edges in the subtree S. Note that z used in (2.5), is completely irrelevant to the random variable z used in the previous sections. Basically, this polynomial is a generating function that encodes the number of subtrees with a given internal and leaf edges [33]. We next show that such polynomials can help us systematically generate trees in a poset from the poset leader. The proof can be found in [31].

Lemma 5. Suppose there is a directed path from the tree T_n to T_{n-m} in a poset, i.e., T_{n-m} can be obtained from T_n through m PGLN-2 operations. Then, their associated polynomials have the following recursive relationship,

$$\begin{aligned} f(T_n;&t,z) = \\ &f(T_{n-m};t,z) + t(1-tz)[m - \sum_{k=1}^{m} g_{n-k}(t,z)] \end{aligned} \tag{2.6}$$

where, $g_{n-k}(t,z)$ is the polynomial associated with the rooted tree obtained from the tree T_{n-k}, after deleting the special leaf edge e and its neighbor edge e' (e.g., see e and e' shown in Figure 2.2 for the tree T_1), in a given step k, and putting their common node as a root (e.g., the node v in Figure 2.2). Note that in (2.6), $T_{n-(n-1)} = T_1$ becomes the LF topology.

Using the recursive equation derived in (2.6), we then have the following corollary, whose proof is in Appendix 2.6.4.

Corollary 1. In a poset, if one of the following cases happens then two polynomials corresponding to the trees are distinct: (1) If there exists a directed path between two trees; (2) If both trees have the same parent tree; (3) If the two structures lie at different levels (stages) in the super-graph.

Hence, by Corollary 1, we see that although Tutte-like polynomial is not graph invariant in general, in many cases the polynomials associated with trees in a same poset are distinct. As an example, consider the poset 5 shown in Figure 2.3. Since all trees satisfy at least one of the conditions in Corollary 1, all of their associated polynomials are thus distinct. Following (2.5), we have

$$\begin{aligned}
f(T_M;t,y) &= t^7y^4 + t^6(y^4+2y^3) + t^5(3y^3+2y^2) \\
&\quad + t^4(4y^2+2y) + t^3(6y+1) + 7t^2 + 7t + 1 \\
f(T_l;t,y) &= t^7y^3 + t^6(3y^3+y^2) + t^5(2y^3+4y^2) \\
&\quad + t^4(5y^2+3y) + t^3(6y+2) + 8t^2 + 7t + 1 \\
f(T_r;t,y) &= t^7y^3 + t^6(3y^3+y^2) + t^5(3y^3+3y^2+y) \\
&\quad + t^4(4y^2+3y+1) + t^3(5y+4) + 9t^2 + 7t + 1 \\
f(T_L;t,y) &= t^7y^2 + 5t^6y^2 + t^5(8y^2+y) \\
&\quad + t^4(6y^2+4y+1) + t^3(5y+5) + 10t^2 + 7t + 1
\end{aligned} \tag{2.7}$$

where T_M and T_L are the MF and LF structures in poset 5, respectively. Also, T_l and T_r are the left and right structures, respectively that located in the middle of poset 5. For the simplicity of polynomials we replaced $z+1$ with y. As we expected, all the computed polynomials in (2.7) are distinct.

The Tutte-like polynomial can be used to evaluate certain topological properties of trees. In the following lemma, whose proof is in [31], we propose an interesting result: the Tutte-like polynomial can enable us to obtain the exact number of special leaf edges in the corresponding

tree. Hence, using this result we estimate the security robustness of a tree structure by computing its distance from LF structure.

Lemma 6. Given the polynomial $f(T;t,z)$ associated with a tree T having $|I|$ internal edges, the second highest degree term has the form $t^{|E|-1}(\alpha(1+z)^{|I|-1}+\beta(1+z)^{|I|})$. The coefficient α shows the number of leaf edges, which are connected to a node with degree two.

Corollary 2. The coefficient α defined in Lemma 6 shows the distance between the tree T and LF structure. Also, if $\alpha = 0$ then T is the LF structure.

Example 1. Consider the tree topologies in poset 5 of Figure 2.3, and their associated polynomials that are computed in (2.7). The MF tree T_M has two special leaf edges, hence in its corresponding polynomial, the second highest degree term has the form $t^6(y^4+2y^3)$. Hence, $\alpha = 2$. The two middle trees in the same poset, each have one special leaf edge, and in this case we have $t^6(3y^3+y^2)$ for both second highest degree terms, in which $\alpha = 1$. On the other hand, the LF tree T_L has no such leaf edges. From (2.7) we can see the second highest degree term for $f(T_L;t,y)$ is $5t^6y^2$, hence $\alpha = 0$.

The results obtained in Lemma 6 and Corollary 2 show a strong correlation between the Tutte-like polynomial and security robustness of a Gaussian tree. In particular, being closer to LF structure, hence having smaller values for α (comparing to others in the same poset), makes the Gaussian tree less favorable comparing to other structures in the same poset.

2.4.2 Enumerating Poset Leaders: Restricted Integer Partition Approach

In the previous sections, we studied certain properties of tree topologies in the same poset. In this section, we find a systematic method to generate different poset leaders, which is further related to *restricted* integer partition problems. The following example will demonstrate new ways to quickly enumerate these MF structures. First consider the following example:

Example 2. Consider the MF structure in poset 5 shown in Figure 2.3. For a moment, picture the node r_5 as a junction to three *branches*. In particular, these branches are chain structures each

having 1, 2, and 4 nodes (excluding r_5), hence we assign the string $(1+2+4)$ to this topology. Note that, each summand in a partition is also called a *part*, e.g., here the parts are 1, 2, and 4. Here, we name r_5 as the *anchor* node to this MF structure. Similarly, in poset 3 and 4 having anchor nodes r_3 and r_4, respectively; we can assign the strings $(1+3+3)$ and $(2+2+3)$ to these structures. The MF structure in the poset 6 has four branches that come out of the anchor node r_6. Therefore, we can assign $(1+2+2+2)$ to this structure. Next, consider the MF topology in poset 1. This is a special case (i.e., an integer partition having two parts), where any internal node can be an anchor node. Here, we arbitrarily choose r_1 as the anchor node, hence obtaining $(3+4)$ for this structure. However, one can choose other internal nodes to obtain equivalent partitions such as $(2+5)$ or $(1+6)$. Note that in all MF structures above we have only one anchor node, hence all the parts in each string sums up to $|V|-1=8-1=7$. Lastly, consider the MF structure in poset 2. Here, there are two anchor nodes r_2 and r_2', each having two branches with lengths 1 and 2. Therefore, the integer partition is separated into two sections, i.e., $(1+2)+(1+2)$, where each section corresponds to one anchor node. In this case since we have two anchor nodes, the parts sum up to $|V|-2=6$.

Based on this example, we propose an effective algorithm to enumerate all poset leaders of a given order. As we anticipate, integer partition methods [22] can be very helpful in order to quickly reach this goal. However, this method should be systematically implemented. In particular, we use *restricted* integer partitions to find all poset leaders. Each integer partition should satisfy the following constraints: (1) Each section should have at most a single 1; (2) The parts in the leftmost (first) and rightmost (last) sections should each sum up to values larger than or equal to 3. Essentially, the first constraint is to ignore the non-poset leader cases, while the second constraint is to ignore the cases where two or more sections can be merged and form already produced sections, hence, making this method more effective. In this case, the partitions that satisfy the above constraints are defined to be *acceptable integer partitions* (AIP). Algorithm 1, effectively finds the list of all AIPs corresponding to poset leaders of given order n:

Input: n, as the order of Gaussian trees
Output: P, as the list of all poset leaders
$P \leftarrow \emptyset$;
for $A := 1$ *to* $A_{\max}$ **do**
  Given $n - A$, find the subset of all AIPs $P_A = \{p_1, p_2, ..., p_{m_A}\}$ each having A parts ;
  for $i := 1$ *to* m_A **do**
    Find those parts in p_i that can be further partitioned to obtain new and AIPs and add them to P_A ;
  end
  Check for any permutation of parts that gives a new AIP and add them to P_A ;
  Check for redundant AIPs in P_A and eliminate them from the list ;
  $P \leftarrow P \cup P_A$;
end

ALGORITHM 1. Enumerating Poset Leaders

Here A shows the number of anchor nodes; and $A_{\max}$ can be determined by combining the two aforementioned constraints. In particular, given $|V| = n$ as the order of trees, then $A_{\max} = n - (3 + 3) = n - 6$. For example, each of the MF structures shown in Figure 2.3 have only one anchor node, except the MF structure in poset 2 that has two anchor nodes, and we know in this case $A_{\max} = 2$. Also, note that unlike normal integer partitions the position of parts matters, so we should count some of permutations of different parts. In particular, two non-isomorphic poset leader topologies may have identical integer partitions, but with different ordering of parts.

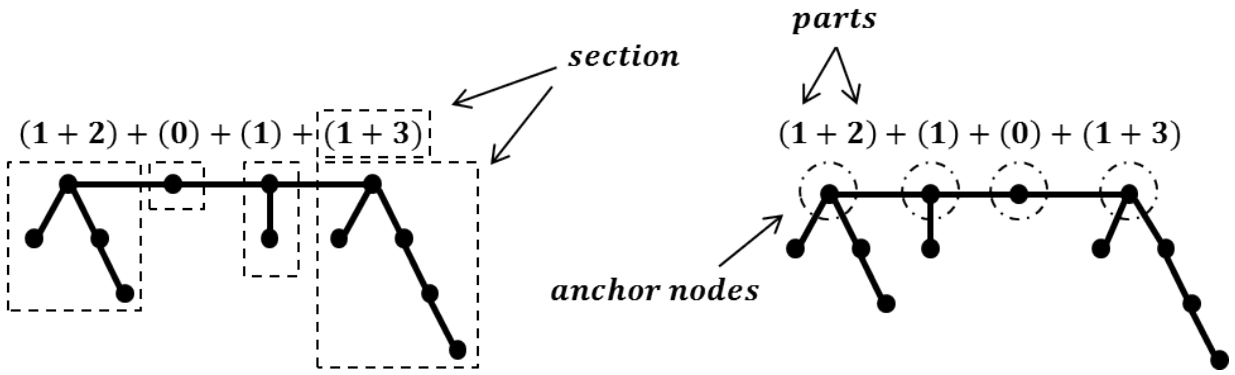


FIGURE 2.4: Two different structures of poset leaders on $n = 12$ nodes

Figure 2.4 shows two different permutations of integer partitions for $n = 12$. As we can see from the figure, these two structures are non-isomorphic, but they have the same parts and sec-

tions. Also, observe that both integer partitions sum up to $(1+2)+(0)+(1)+(1+3) = (1+2)+(1)+(0)+(1+3) = 8$, since based on the algorithm we do not count the anchor nodes (here there are 4 of them) in the partitions.

2.5 Conclusion

In this paper, we studied the problem of comparing security performance of Gaussian trees in both max-min and min-max scenarios. First, we introduced the PGLN-2 operation to obtain an ordering among such trees. The poset of Gaussian trees is defined as equivalence classes containing certain Gaussian trees that can be transformed into each other using one or more PGLN-2 operations. Also, each poset consists of unique MF and LF structures with the best and worst security performances, respectively. Second, we assigned a polynomial to each Gaussian tree, and showed that using such polynomials one can estimate the relative security performance of a Gaussian tree with respect to other structures within the same poset. We also obtained an effective approach, based on restricted integer partitions, to enumerate the MF structures.

2.6 Proof of Theorems

2.6.1 Proof of Lemma 1

First, consider the max-min case, in which Alice and Bob choose a pair of nodes, under the pessimistic assumption that Eve chooses the best possible node to minimize the conditional mutual information $I(a;b|z)$. Figure 2.5 shows all the possible cases that a particular eavesdropper can take, in a fixed path between the nodes a and b. In other words, Eve may pick any node z_1, z_1', z_2, z_2', z_3, z_3' or z_4. Note that there might be several edges on a path between any pair of nodes.

We use the results provided in Theorem 1 and Theorem 2, which are proposed in [21, pp. 348-349]. Essentially, Theorem 1 is a conditional version of the well-known information inequality and holds in general for mutual information of any distribution [30]. Intuitively, for the Gaussian trees the condition in Theorem 1 is satisfied when b lies on the path between a and b', where b' is the alternative choice for Bob. In other words, the longer path implies weaker dependence. On the other hand, Theorem 2 holds in general for the Gaussian joint density. The first part of Theorem

2 shows that if a, b, and z are pairwise separated given x, then conditioning always reduces the mutual information between a and b. In Gaussian trees, the second part of the Theorem 2 shows that for the fixed correlates a and b, the eavesdropper z wants to be closer to the path between them.

From Figure 2.5 we can see that there are totally four possible choices for Eve: When z is connected to the path p_{ab} through one of the nodes a or b; when z is connected to p_{ab} through the node x; and when z lies on the path between a and b.

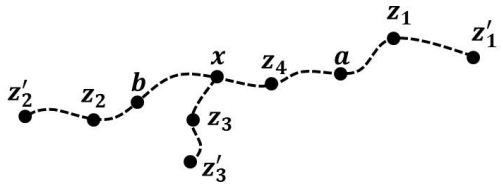


FIGURE 2.5: All the possible locations for the eavesdropper given the fixed correlates

Recall that the objective is to find the value for z that minimizes the mutual information between a and b: $\min_z I(a;b|z)$.

Cases 1 and 2. When z is along the path p_{ab}, i.e., the case z_1 or z_2: First, consider the case z_1, the analysis for z_2 is exactly the same. From Theorem 1 we know that because $a \perp z_1'|z_1$ we have: $I(b;z_1) \geq I(b;z_1')$. Now we want to compare two values for the mutual information. First, observe that $b \perp z_1|a$. So we can conclude that $I(b;a,z_1) = I(b;a)$. The same condition holds for z_1': $I(b;a,z_1') = I(b;a)$.

$$
\begin{aligned}
&I(b;z_1) > I(b;z_1') \rightarrow \\
&I(b;a) - I(b;z_1) < I(b;a) - I(b;z_1') \rightarrow \\
&I(b;a,z_1) - I(b;z_1) < I(b;a,z_1') - I(b;z_1') \rightarrow \\
&I(b;a|z_1) < I(b;a|z_1') \qquad (2.8)
\end{aligned}
$$

Eq. (2.8) shows that $I(a;b|z_1) \leq I(a;b|z_1')$. In other words, the eavesdropper wants to be as close as possible to the path p_{ab}.

Case 3. Now consider the case when z is a branch node, i.e., it is connected to p_{ab} through the node x: It is obvious that by replacing z_3 with z and z_3' with z' in the Theorem 2's conditions, we can satisfy all the constraints in this theorem. Hence, we can conclude that $I(a;b|z_3) \leq I(a;b|z_3')$. Again, we conclude that z wants to be closer to the path p_{ab}.

Case 4. When z lies on the path p_{ab}: In this case it is obvious that $a \perp b|z_4$. As a result we have $I(a;b|z_4) = 0$, which is not desirable choice for a and b.

Next, we find possible cases that maximize the mutual information between a and b, given the fixed node for z. We show that to maximize the conditional mutual information, a and b should be close to each other. Consider the case where Bob has two choices b or b', where b is on the path between b' and Alice's choice a. Then for any given subset of choices Z for Eve, by data processing inequality [30] we have $I(a;b|Z) \geq I(a;b'|Z)$ Hence, we can immediately see that Alice and Bob pick the pair of nodes that are adjacent. Also, it can be argued that if a and b are not adjacent, then the eavesdropper wants to pick the best node: z picks any node on the path p_{ab}. As a result $I(a;b|z)$ becomes zero. This validates the first result in Lemma 1.

Next, consider the min-max case, in which Eve chooses a particular node, assuming that Alice and Bob choose the best pair of nodes to maximize the conditional mutual information $I(a;b|z)$. Similar to the max-min case, observe that regardless of Eve's choice, Alice and Bob choose adjacent nodes, since otherwise based on Theorem 1, $I(a;b|z)$ becomes either zero, or it can be improved further. Furthermore, let us assume that Eve picks a leaf node, say z, which is adjacent to z'. Now, the min-max value for this particular case is computed by $\rho^2_{ab|z} = max_{(a,b)\in E\setminus(z',z)}$. On the other hand, if the eavesdropper picks z', the min-max value becomes $\rho'^2_{ab|z'} = max_{(a,b)\in E\setminus(z',adj(z'))}$, where $adj(z')$ is the set of adjacent nodes to z', which contains z as well as some other nodes. Hence, using Theorem 2 clearly $\rho'^2_{ab|z'} \leq \rho^2_{ab|z}$ and since Eve chooses the minimum between all possible cases, so it rules out all the leaf nodes. This completes the proof.

2.6.2 Proof of Lemma 2

We first prove the following:

For any $v_i \in V$, let us define d_i as its degree. Then, we have,

$$|\Sigma_T| = \frac{\prod\limits_{(v_i,v_j)\in E} [\sigma_{v_i v_i}\sigma_{v_j v_j} - \sigma^2_{v_i v_j}]}{\prod\limits_{v_i \in V} \sigma^{d_i-1}_{v_i v_i}} \tag{2.9}$$

The proof follows by induction. First, assume that the Gaussian tree T has only one edge with two vertices v_1 and v_2, then we can immediately form Σ_T, and deduce $|\Sigma_T| = \sigma_{v_1 v_1}\sigma_{v_2 v_2} - \sigma^2_{v_1 v_2}$, which follows the general formula in (2.9). Next, let us assume that (2.9) is valid up to $T' = (V', E', W')$, where $|V'| = n - 1$. Hence, we need to prove the validity of this equation for $T = (V, E, W)$ with $|V| = n$, where the tree T can be obtained from T' by adding one vertex, namely v_n. Without loss of generality, we assume that v_n is connected to v_{n-1}. Since $v_n \perp v_i | v_{n-1}$, for all $v_i \in V\backslash\{v_n, v_{n-1}\}$, using the arguments given in Section 2.2, we have $\sigma_{v_i v_n} = \sigma_{v_i v_{n-1}}\sigma_{v_{n-1} v_n}/\sigma_{v_{n-1} v_{n-1}}$. If we factorize $\sigma_{v_{n-1} v_n}/\sigma_{v_{n-1} v_{n-1}}$ from the last column, then subtract the $n-1$-th column from the n-th column, and replace the result with the n-th column, we obtain,

$$|\Sigma_T| = \frac{\sigma_{v_{n-1} v_n}}{\sigma_{v_{n-1} v_{n-1}}} \begin{vmatrix} \sigma_{v_1 v_1} & \cdots & 0 \\ \sigma_{v_2 v_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ \sigma_{v_{n-1} v_1} & \cdots & 0 \\ \sigma_{v_n v_1} & \cdots & x \end{vmatrix}$$

where $x = (\sigma_{v_n v_n}\sigma_{v_{n-1} v_{n-1}} - \sigma^2_{v_n v_{n-1}})/\sigma_{v_n v_{n-1}}$. Using the last column, we can compute $|\Sigma_T|$ as follows,

$$\begin{aligned} |\Sigma_T| &= \frac{\sigma_{v_{n-1} v_n}}{\sigma_{v_{n-1} v_{n-1}}} x |\Sigma_{\backslash n \backslash n}| \\ &= \frac{\sigma_{v_n v_n}\sigma_{v_{n-1} v_{n-1}} - \sigma^2_{v_n v_{n-1}}}{\sigma_{v_{n-1} v_{n-1}}} |\Sigma_{\backslash n \backslash n}| \end{aligned} \tag{2.10}$$

where $|\Sigma_{\backslash n \backslash n}|$ is the determinant of submatrix of Σ_T resulting after removing the n-th column and row. Note that since removing the last row and column of Σ_T is the same as removing v_n from T,

hence we conclude that $\Sigma_{\backslash n \backslash n} = \Sigma_{T'}$. Therefore, (2.10) becomes

$$|\Sigma_T| = \frac{\sigma_{v_n v_n}\sigma_{v_{n-1}v_{n-1}} - \sigma^2_{v_n v_{n-1}}}{\sigma_{v_{n-1}v_{n-1}}}|\Sigma_{T'}| \tag{2.11}$$

Observe that since the degree of v_{n-1} is 2, the fraction in (2.11) has the same form as in (2.9). Also, we know that $|\Sigma_{T'}|$ follows the general formula as well. This completes the inductive proof for the first part.

To prove the result in Lemma 2, note that the Gaussian trees T_1 and T_2 differ in only one edge, namely e. Hence, we can write,

$$|\Sigma_{T_1}| = |\Sigma_{\backslash n_2 \backslash n_2}|\frac{\sigma_{n_1 n_1}\sigma_{n_2 n_2} - \sigma^2_{n_1 n_2}}{\sigma_{n_1 n_1}}$$
$$|\Sigma_{T_2}| = |\Sigma_{\backslash n_2 \backslash n_2}|\frac{\sigma_{v' v'}\sigma_{n_2 n_2} - \sigma^2_{v' n_2}}{\sigma_{v' v'}}$$

Since we assume that $|\Sigma_{T_1}| = |\Sigma_{T_2}|$, the result follows.

2.6.3 Proof of Lemma 3

First, note that since PGLN-2 changes the local structure, most of the parts in both trees T_1 and T_2 shown in Figure 2.2 remains the same. This in turn results in both max-min values to be equal in many cases. Let us denote the squared partial correlation coefficients for trees T_1 and T_2 as $\rho^2_{ab|z}$ and $\rho'^2_{a'b'|z'}$, respectively, then we have the following cases for the max-min scenario:

1. Suppose in tree T_1, Alice and Bob choose a pair $(a, b) \in E_C(T_1)$, where $E_C(T_1)$ is the set of all edges inside the cloud other than v. Then, according to Lemma 1, Eve chooses z from appropriate nodes in $V_C(T_1)$, i.e., the set of nodes inside the cloud (including v). Now, if $(a', b') \in E_C(T_2)$ then since $E_C(T_2) = E_C(T_1)$ and $V_C(T_2) = V_C(T_1)$, so the max-min values for this case are equal.

2. Suppose in tree T_1, Alice and Bob choose the pair $(a, b) = (x_i, v)$, where $x_i \in adj(v)$ is adjacent to v. Then $z \in \{adj(x_i), n_1\}$. Now, if in T_2 the pair $(a', b') \in (x_i, v)$, then $z' \in \{adj(x_i), n_1, n_2\}$. In T_2, Eve has one more option (i.e., n_2) to choose from, comparing to its choices in T_1, hence we can immediately conclude that for this case $\rho^2_{ab|z} \geq \rho'^2_{a'b'|z'}$.

3. Suppose in T_1, Alice and Bob choose the pair $(a, b) = (v, n_1)$. In this case $z \in \{adj(v), n_2\}$. Now, if in T_2 the pair $(a', b') = (v, n_1)$, then $z' \in \{adj(v), n_2\}$. Now we know $\sigma^2_{n_1n_2}/\sigma_{n_1n_1} = \sigma'^2_{vn_2}/\sigma_{vv}$, then if we replace $\sigma'^2_{vn_2}$ with $\sigma^2_{n_1n_2}\sigma_{vv}/\sigma_{n_1n_1}$ in the equation regarding to $\rho'^2_{vn_1|n_2}$ we can conclude that $\rho^2_{vn_1|n_2} = \rho'^2_{vn_1|n_2}$. As a result, the max-min values for both trees in this case are equal.

4. Suppose in T_1, Alice and Bob choose the pair $(a, b) = (n_1, n_2)$. Then Eve has only one option, which is choosing v. Now, if in T_2, $(a', b') = (v, n_2)$, then $z' \in adj(v)$, where $adj(v)$ consists of the set of vertices inside the cloud, as well as n_1. Now, using similar arguments as in case 3, and by $\sigma^2_{n_1n_2}/\sigma_{n_1n_1} = \sigma'^2_{vn_2}/\sigma_{vv}$, we can show that $\rho^2_{n_1n_2|v} = \rho'^2_{vn_2|n_1}$. Since, in T_2, Eve can choose any z' other than n_2, hence in this case $\rho^2_{ab|z} \geq \rho'^2_{a'b'|z'}$.

Following discussed cases, showing that $T_1 \succeq T_2$ is straightforward: for example, suppose $S_M(T_1, W)$ is chosen from case 1, then if $S_M(T_2, W)$ is chosen from the same case, we know $S_M(T_1, W) = S_M(T_2, W)$. Otherwise, if $S_M(T_2, W)$ is chosen from any other case (let's name this value as $S'_M(T_2, W)$), then since Eve chooses the minimum among the four cases, so $S'_M(T_2, W) \leq S_M(T_2, W) = S_M(T_1, W)$. As another example, suppose $S_M(T_1, W)$ is chosen from case 2, then if $S_M(T_2, W)$ is chosen from the same case, we know $S_M(T_1, W) \geq S_M(T_2, W)$. Otherwise, if the max-min value, say $S'_M(T_2, W)$ is chosen from any other case, using the same arguments $S'_M(T_2, W) \leq S_M(T_2, W) \leq S_M(T_1, W)$. Similar arguments can be used for the remaining cases.

Next, similar to the max-min problem, we can conclude the following cases for the min-max problem:

1. Suppose in T_1, the eavesdropper picks a (non-leaf) node v_c from inside the cloud, where $v_c \in V_C(T_1)$. Then, the possible choices for the pair Alice and Bob are $(a, b) \in \{E_C(T_1), (v, n_1), (n_1, n_2)\}$. If we also assume $z' = v_c$, then $(a', b') \in \{E_C(T_2), (v, n_1), (v, n_2)\}$. We know since $E_C(T_1) = E_C(T_2)$, hence the only difference is the pair $(n_1, n_2) \in E(T_1)$ versus $(v, n_2) \in E(T_2)$. Using the fact that $\sigma^2_{n_1n_2}/\sigma_{n_1n_1} = \sigma'^2_{vn_2}/\sigma_{vv}$ and using (2.4) it is not hard to show that $\rho^2_{n_1n_2|v_c} \geq \rho'^2_{vn_2|v_c}$ for all $v_c \in V_C(T_1)$. As a result, for this case we have $\rho^2_{ab|z} \geq \rho'^2_{a'b'|z'}$.

2. Suppose in T_1, the eavesdropper picks the node v. Then, for the pair of legitimate nodes we have $(a, b) \in \{E_C(T_1), (n_1, n_2)\}$. If we also assume that $z' = v$, then $(a', b') \in E_C(T_2)$. Now, we know the Alice and Bob want to maximize their security; since in T_1 they have one more option (i.e., (n_1, n_2)) to choose from, therefore for this case again we have $\rho^2_{ab|z} \geq \rho'^2_{a'b'|z'}$.

3. Suppose in T_1, Eve picks the node n_1. Then, $(a, b) \in E_C(T_1)$. Note that in the tree T_2 the node $z' \neq n_1$, since it is a leaf. Hence, again suppose $z' = v$. So, similar to case 2 we know $(a', b') \in E_C(T_2)$. Note that the node z' lies on the path from the pairs (a', b') (inside the cloud in T_2) to n_1. Therefore, using Theorem 2 in [21, p. 349] we conclude that $\rho^2_{ab|z=n_1} \geq \rho'^2_{a'b'|z'=v}$.

Now, we show $T_1 \succeq T_2$: for example, suppose $S_M(T_1, W)$ is chosen from case 1, then if $S_M(T_2, W)$ is chosen from the same case, we know $S_M(T_1, W) \geq S_M(T_2, W)$. Otherwise, if the max-min value, say $S'_M(T_2, W)$ is chosen from the other case (i.e., case 2), since (a, b) in T_1 have chosen the case with maximum value $S_M(T_1, W) \geq \rho^2_{ab|z=n_1}$, where $\rho^2_{ab|z=n_1}$ corresponds to case 3 in T_1. But we know from above that $\rho^2_{ab|z=n_1} \geq \rho'^2_{a'b'|z'=v} = S'_M(T_2, W)$, hence $S_M(T_1, W) \geq S'_M(T_2, W)$. We can use similar arguments for the other cases as well. This completes the proof.

2.6.4 Proof of Corollary 1

First, suppose $f(T_n; t, z) = f(T_{n-m}; t, z)$ then using (2.6) we should have $t(1 - tz)\ [m - \sum_{k=1}^{m} g_{n-k}(t, z)] = 0$, or $\sum_{k=1}^{m} g_{n-k}(t, z) = m$. Recall that all $g_{n-k}(t, z)$ are polynomials associated with rooted trees, so the only possibility is $g_{n-k}(t, z) = 1$, for all $1 \leq k \leq m$, a contradiction.

Second, consider two trees T_{n-m} and T_{n-l}, at different levels having nearest common ancestor T_n. Then using (2.6) we have the following:

$$f(T_n; t, z) = \begin{cases} f^L(T_{n-m}; t, z) + t(1 - tz)[m - \sum_{k=1}^{m} g^L_{n-k}] \\ f^R(T_{n-l}; t, z) + t(1 - tz)[l - \sum_{k=1}^{l} g^R_{n-k}] \end{cases}$$

suppose, $f^L(T_{n-m}; t, z) = f^R(T_{n-l}; t, z)$ then we obtain,

$$\sum_{k=1}^{m} g^L_{n-k} - \sum_{k=1}^{l} g^R_{n-k} = m - l \tag{2.12}$$

If $m = l = 1$, i.e., both trees T_{n-m} and T_{n-l} are obtained from T_n by a single grafting operation. But, since they have two different structures, the corresponding polynomials for the rooted trees g^L_{n-1} and g^R_{n-1} are distinct, because in [13] it is shown that Tutte-like polynomial for rooted trees is graph invariant. Hence, $f^L(T_{n-1}; t, z)$ and $f^R(T_{n-1}; t, z)$ are distinct. So, the trees at the same level that are obtained from their parent through one grafting operation are distinct.

Finally, suppose we have $m \neq l$. Let's define $y_i = g_i - 1$ for all polynomials $g_i(t, z)$. Now, using (2.12) we have,

$$\sum_{k=1}^{m} y^L_{n-k} - \sum_{k=1}^{l} y^R_{n-k} = 0 \tag{2.13}$$

The highest degree term corresponds to the rooted trees resulted by eliminating the edges e and e' and putting the common node between these two edges as a root. Also, the highest degree terms are resulted from the subtrees associated to y^L_i and y^R_i and no other proper subsets of these trees. Hence, from (2.13) and assuming that original tree T_n has the size $|E|$, then we can conclude,

$$t^{|E|-2} \sum_{k=1}^{m} (1 + z)^{L_{n-k}} = t^{|E|-2} \sum_{k=1}^{l} (1 + z)^{R_{n-k}} \tag{2.14}$$

where L_{n-k} and R_{n-k} are non-negative integer powers, which show the largest number of internal edges for each tree associated to polynomials y^L_{n-k} and y^R_{n-k}.

Equation (2.14) should hold for all values of t and z. Let's set $t = 1$ and $z = 0$, we obtain $m = l$, a contradiction.

Chapter 3

Layered Synthesis of Latent Gaussian Trees

In this chapter, a new synthesis scheme is proposed to generate a random vector with prescribed joint density that induces a (latent) Gaussian tree structure. The quality of synthesis is shown by vanishing total variation distance between the synthesized and desired statistics. The proposed layered and successive synthesis scheme relies on the learned structure of tree to use sufficient number of common random variables to synthesize the desired density. We characterize the achievable rate region for the rate tuples of multi-layer latent Gaussian tree, through which the number of bits needed to synthesize such Gaussian joint density are determined. The random sources used in our algorithm are the latent variables at the top layer of tree, the additive independent Gaussian noises, and the Bernoulli sign inputs that capture the ambiguity of correlation signs between the variables. We have shown that such ambiguity can further help in reducing the synthesis rates for the underlying Gaussian trees.

This study has resulted in three research papers, [34–36].

3.1 Introduction

Consider the problem of simulating a random vector with prescribed joint density. Such *generative modeling* can be implemented by generating an appropriate number of random input bits (by relying on a random source) to a stochastic channel whose output vector has its empirical statistics meeting the desired one measured by a given metric. Generative models have many applications ranging from probabilistic programs [37] to economics [38], physics [39] and computer vision [40].

We aim to address such synthesis problem for a case where the prescribed output statistics induces a (latent) *Gaussian tree* structure, i.e., the underlying structure is a tree and the joint density of the variables is captured by a Gaussian density. The Gaussian graphical models are widely

studied in the literature, because of a direct correspondence between conditional independence relations occurring in the model with zeros in the inverse of covariance matrix, known as the *concentration matrix*. They have diverse applications in social networks, biology, and economics [41, 42], to name a few. Gaussian trees in particular have attracted much attention [42] due to their sparse structures, as well as existing computationally efficient algorithms in learning the underlying topologies [2, 43].

In a latent Gaussian tree, we are dealing with two sets of variables. Let $\mathbf{X} = \{X_1, X_2, ..., X_n\}$ be the n observed variables in a Gaussian tree, i.e., the covariance matrix $\Sigma_{\mathbf{x}}$ is given. The set of variables $\mathbf{Y} = \{Y_1, Y_2, ..., Y_k\}$ are hidden to us and should be estimated. Note that for $\Sigma_{\mathbf{x}}$ to induce a latent Gaussian tree, it needs to satisfy certain conditions shown in [43]. In fact, for any triplet $x_i, x_j, x_k \in \mathbf{X}$ and writing $\rho_{x_i x_j}$ to show the pairwise correlation we need to have $|\rho_{x_i x_j}| \geq |\rho_{x_i x_k} \rho_{x_j x_k}|$ and $\rho_{x_i x_j} \rho_{x_i x_k} \rho_{x_j x_k} > 0$. Such constraints on the correlation space shown to be necessary and sufficient for a joint Gaussian distribution to characterize a latent Gaussian tree [43].

There are several works such as [2, 44] that have proposed efficient algorithms to infer the latent Gaussian tree parameters. In fact, Choi et al., proposed a new *recursive grouping* (RG) algorithm along with its improved version, i.e., *Chow-Liu* RG (CLRG) algorithm to recover a latent Gaussian tree that is both *structural* and *risk* consistent [2], hence it recovers the *correct* value for the latent parameters. They introduced a *tree metric* as the negative *log* of the absolute value of pairwise correlations to perform the algorithm.

In this chapter we assume that the parameters and structure information of the latent Gaussian tree is provided using one of aforementioned algorithms.

Our primary concern in such synthesis problem is about efficiency in terms of the amount of random bits required at the input, as well as the modeling complexity of given stochastic system through which the Gaussian vector is synthesized. Such efficiency is characterized through defining proper random *sequences*, and random *bins* containing those sequences, which we define

as random *codewords* and *codeboooks*. We use the input code-rate to define the complexity of our synthesis systems, since minimizing such rates results in reducing the number of common random bits needed to generate the output statistics. In particular, through showing the intrinsic *sign singularity* in latent Gaussian trees, we have demonstrated that such ambiguity can further help us to reduce the synthesis rates for such Gaussian trees. To clarify, we consider the following case study.

3.1.1 Motivating Case Study

Consider a Gaussian tree shown in Figure 3.1. It consists of four observed variables X_1, X_2, X_3, and X_4 that are connected to each other through two hidden nodes $Y_1^{(1)}$ and $Y_2^{(1)}$. Define $\rho_{x_1y_1} = E[X_1Y_1^{(1)}]$ as the true correlation value (edge-weight) between the input $Y_1^{(1)}$ and the output X_1. We can similarly define other correlation values $\rho_{x_2y_1}$, $\rho_{x_3y_2}$, and $\rho_{x_4y_2}$. Define $B_j^{(1)} \in \{-1, 1\},\ j \in [1, 2]$ as a binary variable corresponding to the $j-th$ input that as we will see reflects the sign information of pairwise correlations. For the tree shown in Figure 3.1, one may assume that $B_j^{(1)} = 1$ to show the case with $\rho'_{x_iy_j} = \rho_{x_iy_j}$, while $B_j^{(1)} = -1$ captures $\rho''_{x_iy_j} = -\rho_{x_iy_j}$, where $\rho'_{x_iy_j}$ and $\rho''_{x_iy_j}$, $i \in [1, 2]$ *or* $i \in [3, 4]$ are the (alternative) recovered correlation values using certain inference algorithm such as RG [2]. Also, define $B_{12} = B_1^{(1)}B_2^{(1)}$. It is easy to see that both recovered correlation values induce the same covariance matrix $\Sigma_{\mathbf{x}}$, showing the sign singularity issue in such a latent Gaussian tree. In particular, for each pairwise correlation $\rho_{x_kx_l}, k < l \in [1, 2, 3, 4]$, and if x_k and x_l have the same parent, we have $\rho_{x_kx_l} = \rho_{x_ky_j}\rho_{x_ly_j} = (B_j^{(1)})^2\rho_{x_ky_j}\rho_{x_ly_j}$, where the second equality is due to the fact that regardless of the sign value, the term $(B_j^{(1)})^2$ is equal to 1. Now, depending on whether we replace $B_j^{(1)}$ with $\{1, -1\}$, we obtain $\rho_{x_kx_l} = \rho'_{x_ky_j}\rho'_{x_ly_j} = \rho''_{x_ky_j}\rho''_{x_ky_j}$. And there is no way to distinguish these two groups using only the given information on observables joint distribution. Similarly, if x_k and x_l are connected to different input nodes, we can write $\rho_{x_kx_l} = \rho_{x_ky_1}\rho_{x_ly_2} = B_1^{(1)}B_2^{(1)}B_{12}\rho_{x_ky_j}\rho_{x_ly_j}$, where the second equality is due to $B_{12} = B_1^{(1)}B_2^{(1)}$. Again, one cannot recover the sign information from only the output correlation values.

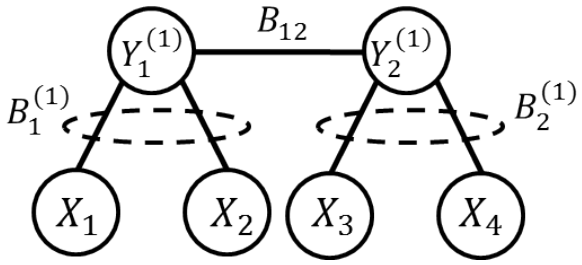


FIGURE 3.1: A simple Gaussian tree with a hidden node $Y^{(1)}$

Such sign singularity patterns become more complex as the tree size grows. In section 3.3 we characterize certain properties of sign information.

It turns out that such sign singularity can be seen as another *noisy* source of randomness, which can further help us to reduce the code-rate corresponding to latent inputs to synthesize the latent Gaussian tree. In fact, we may think of the Gaussian tree shown in Figure 3.1 as a communication channel, where information flows from the source $\mathbf{Y}^{(1)} = [Y_1^{(1)}, Y_2^{(2)}]$ through four channels $p_{X_i|Y_j^{(1)}}$ with independent additive Gaussian noise variables $Z_i \sim N(0, \sigma_{z_i}^2),\ i \in \{1,2,3,4\}$ to generate (dependent) outputs with $\mathbf{X} \sim N(0, \Sigma_{\mathbf{x}})$. We introduce $\mathbf{B}^{(1)} = [B_1^{(1)}, B_2^{(2)}] \in \{-1, 1\}$ as binary Bernoulli random variables with parameters $\pi_{\mathbf{B}^{(1)}}$ and $\pi_{\mathbf{B}^{(2)}}$, which reflect the sign information of pairwise correlations. In fact, we may define the following affine transformation from inputs to outputs,

$$\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} = \begin{bmatrix} \alpha_{11} B_1^{(1)} & 0 \\ \alpha_{21} B_1^{(1)} & 0 \\ 0 & \alpha_{32} B_2^{(1)} \\ 0 & \alpha_{42} B_2^{(1)} \end{bmatrix} \begin{bmatrix} Y_1^{(1)} \\ Y_2^{(1)} \end{bmatrix} + \begin{bmatrix} Z_1 \\ Z_2 \\ Z_3 \\ Z_4 \end{bmatrix} \tag{3.1}$$

where α_{ij} are given values that characterize the correlations up to sign, i.e., $|\rho_{x_i y_j}|$.

Our goal is to characterize the achievable rate region and through a synthesis scheme to generate Gaussian trees with density $q_{\mathbf{XY}^{(1)}}$ using only the hidden inputs and through a channel with additive Gaussian noises, where the synthesized joint density $q_{\mathbf{XY}^{(1)}}$ is indistinguishable from the true Gaussian tree density $p_{\mathbf{XY}^{(1)}}$ as measured by *total variation* metric [16]. To achieve this,

we first generate many sampled sequences (codewords) $(\mathbf{Y}^{(1)})^N$ and $(\mathbf{B}^{(1)})^N$ to form the corresponding bins (codebooks) containing those codewords. The size of the codebooks are characterized by $2^{NR_{\mathbf{Y}^{(1)}}}$ and $2^{NR_{\mathbf{B}^{(1)}}}$, where $R_{\mathbf{Y}^{(1)}}$ and $R_{\mathbf{B}^{(1)}}$ are the codebook rates, regarding to sign and hidden nodes codewords. Each time to generate output sequences, we first randomly pick sign and latent node codewords and then we use the synthesis channel in (3.1) to achieve a particular output sequence $\mathbf{X}^N$. We aim to characterize the lower bound on the codebook rates, through which the generated sequence's statistics, i.e., $q_{\mathbf{X}\mathbf{Y}^{(1)}}$ is asymptotically (as $N \to \infty$) indistinguishable from the desired statistics. In particular, we characterize the quantity $\inf_{p_{\tilde{\mathbf{Y}}^{(1)}}} I(\mathbf{X}; \tilde{\mathbf{Y}}^{(1)})$, where $I(\mathbf{X}; \tilde{\mathbf{Y}}^{(1)})$ is the mutual information between the output $\mathbf{X}$ and the input vector $\tilde{\mathbf{Y}}^{(1)} = \{\mathbf{Y}^{(1)}, \mathbf{B}\}$. This corresponds to finding the minimum achievable rate under Gaussian tree assumption. Equivalently, we are seeking for optimal values of $\pi_{\mathbf{B}^{(1)}}$ and $\pi_{\mathbf{B}^{(2)}}$ to maximize the achievable rate region characterized by $R_{\mathbf{Y}^{(1)}}$ and $R_{\mathbf{B}^{(1)}}$.

Remark 1. Suppose for a moment, instead of using the tree structure, we simply used six independent normalized Gaussian variables and by passing them through a filter, i.e., linear combination of these independent variables, we produce the desired Gaussian tree (with two hidden nodes and fours observables). While this approach seems appealing, note that as it is observed in [45] as well, such synthesis scheme needs infinite bits of precision to produce the desired statistics, which is practically infeasible. This is due to the noiseless nature of the channel (see *channel resolvability* [15]), i.e., the linear filter, which is noise-free that makes the input code rates maximized (since the input-output mutual information will be maximized), hence, we need infinite bits of precision to synthesize the desired Gaussian density. In contrast, our framework exploits the tree structure to further reduce the rates needed for synthesis. Moreover, to characterize the channel shown in Figure 3.1, one may need to introduce only four parameters α_{ij}, one for each edge, while the aforementioned *naive* approach needs nine parameters (basically each input is connected to all the outputs) to capture the dependency structure of output variables. This modeling efficiency will become more evident in more general and larger Gaussian trees, since in that case the naive

approach faces with $\mathcal{O}(n^2)$ parameters while our approach only needs $\mathcal{O}(n+k)$ parameters (in the order of edge-set cardinality of the tree), where n and k are the number of outputs, and latent inputs, respectively. Such efficiency is captured by sparsity structure of connection matrix A_B between the input and output, which will be completely characterized in subsequent sections.

3.1.2 Related Works

Wyner's Common information characterizes the minimum amount of common randomness needed to approximate the joint density between a pair of random variables X_1 and X_2 to be $C(X_1, X_2) = \min_{\substack{P_Y \\ X_1 - Y - X_2}} I(X_1, X_2; Y)$, where $C(X_1, X_2)$ is widely known as *Wyner's common information*. This is done through a common source of randomness, i.e., Y, and two independent random sources to generate X_1 and X_2 with desired joint statistics. Han and Verdu, in [15] along the same problem, define the notion of *resolvability* of a given channel, which is defined as the minimal required randomness to generate output statistics in terms of a vanishing total variation distance between the synthesized and prescribed joint densities. Resolvability of a channel is found to be a very intuitive description of common randomness in our settings, since it can be related to channel quality in terms of its noise power, and the noisier the channel the less number of common random bits needed to simulate the output [15]. Along the same line, Cuff in [16] completely characterized the achievable rate regions needed to synthesize a memoryless channel, where he also used the total variation distance metric to show the quality of the proposed scheme.

There are several works that extend the classical bi-variate synthesis problem in Wyner's study to more general scenarios. In [46–48], the authors aim to define the common information of n dependent random variables, to further address the same question in this setting. A lower bound on such generalized common information is obtained in [49]. Also, the common information for a special case with n Gaussian variables with homogeneous pairwise correlations is obtained. They resort to the same scenario as Wyner [14] did, i.e., considering one random variable to define such common randomness. Veld and Gastpar [50] characterize such quantity for a more general set of Gaussian vectors with circulant covariance matrices. Also, in [51] the authors completely

characterize the common information between two jointly Gaussian vectors, as a function of certain singular values that are related to both joint and marginal covariance matrices of two Gaussian random vectors. However, they still divide the random vector into two groups, which makes it similar to Wyner's scenario.

In this chapter, we are not concerned with solving the common information problem for Gaussian trees. Instead, we want to motivate the notion *multi-variable* synthesis, that is instead of introducing a single variable Y, we define a random vector $\mathbf{Y}$ with certain dependency structure to capture the common randomness and produce common random bits. We provide a layered synthesis algorithm, along with the corresponding achievability regions to synthesize those distributions inducing a Gaussian tree. In [52] such general case is appropriately defined using a constrained convex optimization problem. The benefits of such general assumption is shown in [45]. In fact, Steeg *et. al.* implement a new method based on *multi-layer* common variables for a particular blind source separation problem and showed that their proposed model outperforms all previous learning algorithms.

Similar to [45, 52] we also consider multi-variable cases, but unlike those works, we are interested in characterizing the achievable rates to synthesize a special class of Gaussian distributions, namely Gaussian trees. We adopt a specific (but natural) structure to our synthesis scheme to decrease the number parameters to model the synthesis scheme. It is worthy to point that the achievability results are under the assumed structured synthesis framework. Hence, although through defining an optimization problems, we show that the proposed method is efficient in terms of both modeling and codebook rates, the converse proof, which shows the optimality of such scheme and rate regions is never claimed.

3.1.3 Contributions

Our main contributions can be summarized as follows:

• We propose a novel generative modeling scheme, by which we synthesize any Gaussian vector that lies in a subspace of latent Gaussian trees. The proposed scheme is modeling-wise

efficient, since by relying on the inferred latent tree structure it reduces the number of parameters needed at each step for output synthesis. We also characterize the achievable rate regions for all the channels at each layer.

• We prove that under the latent Gaussian tree assumption, the mutual information between the output vector and both latent inputs and sign variables is only a function of output's covariance matrix $\Sigma_{\mathbf{X}}$. We provide a general formula for such mutual information in a case of *leaf* outputs. We also show that given the sign information, the mutual information between each adjacent layer vectors is fixed as well. We show that the achievable rates are lower bounded by input-output mutual information values at each layer.

• We show that the lower bounds on latent variable rates are a function of Bernoulli sign variables. Such sign ambiguity can be seen as another source of randomness to further help us achieve lower codebook rates for synthesis. We prove that such lower bounds can be minimized (hence maximizing the achievable rate region) in a case of homogeneous Bernoulli distributed sign information.

• In our previous work [53], we only characterized the achievable rate regions for output synthesis of the latent Gaussian trees with leaf observables, and with each hidden node only connected to the upper layer inputs. However, here not only we provide a constructive proof for those subclass of Gaussian trees, but also we completely characterize the synthesis scheme to generate the entire statistics of any latent Gaussian tree structure.

3.2 Problem Formulation

3.2.1 The Signal Model of a Multi-Layer Latent Gaussian Tree

Here, we suppose a latent graphical model, with $\mathbf{Y} = [Y_1, Y_2, ..., Y_k]'$ as the set of inputs (hidden variables), $\mathbf{B} = [B_1, ..., B_m]$, with each $B_i \in \{-1, 1\}$ being a binary Bernoulli random variable with parameter $\pi_i = p(B_i = 1)$ to introduce sign variables, and $\mathbf{X} = [X_1, X_2, ..., X_n]'$ as the set of Gaussian outputs (observed variables) with $p_{\mathbf{X}}(\mathbf{x})$. We also assume that the underlying network structure is a latent Gaussian tree, therefore, making the joint probability (under each sign

realization) $p_{\mathbf{XY|B}}$ be a Gaussian joint density $N(\mu, \Sigma_{\mathbf{xy|b}})$, where the covariance matrix $\Sigma_{\mathbf{xy|b}}$ induces tree structure $G_T(V, E, W)$, where V is the set of nodes consisting of both vectors $\mathbf{X}$ and $\mathbf{Y}$; E is the set of edges; and W is the set of edge-weights determining the pairwise covariances between any adjacent nodes. We consider normalized variances for all variables $X_i \in \mathbf{X},\ i \in \{1, 2, ..., n\}$ and $Y_j \in \mathbf{Y},\ j \in \{1, 2, ..., k\}$. Such constraints do not affect the tree structure, and hence the independence relations captured by $\Sigma_{\mathbf{xy|b}}$. Without loss of generality, we also assume $\mu = \mathbf{0}$, this constraint does not change the amount of information carried by the observed vector.

In [53] we showed that the vectors $\mathbf{X}$ and $\mathbf{B}$ are independent, and the intrinsic sign singularity in Gaussian trees is due to the fact that the pairwise correlations $\rho_{x_i x_j} \in \Sigma_{\mathbf{x}}$ can be written as $\prod_{(l,k)\in E} \rho_{x_l x_k}$, i.e., the product of correlations on the path from x_i to x_j. Hence, roughly speaking, one can carefully change the sign of several correlations of the path, and still maintain the same value for $\rho_{x_i x_j}$. Although this results in no variation on the correlation values $\rho_{x_n x_m},\ n, m \in V$, we showed that if the cardinality of the input vector $\mathbf{Y}$ is k, then 2^k minimal Gaussian trees (that only differ in sign of pairwise correlations) may induce the same joint Gaussian density $p_{\mathbf{X}}$ [53].

In order to propose the successive synthesis scheme, we need to characterize the definition of *layers* in a latent Gaussian tree. We define latent vector $\mathbf{Y}^{(l)}$, to be at layer l, if the shortest path between each latent input $Y_i^{(l)} \in \mathbf{Y}^{(l)}$ and the observed layer (consisting the output vector $\mathbf{X}$) is through l edges. In other words, beginning from a given latent Gaussian tree, we assume the output to be at layer $l = 0$, then we find its immediate latent inputs and define $\mathbf{Y}^{(1)}$ to include all of them. We iterate such procedure till we include all the latent nodes up to layer L, i.e., the top layer. In such setting, the sign input vector $\mathbf{B}^{(l)}$ with Bernoulli sign random variables $B_i^{(l)} \in \mathbf{B}^{(l)}$ is assigned to the latent inputs $\mathbf{Y}^{(l)}$.

We adopt a synthesis channel to feature the relationship between each pair of successive layers. Assume $\mathbf{Y}^{(l+1)}$ and $\mathbf{B}^{(l+1)}$ as the input vectors, $\mathbf{Y}^{(l)}$ as the output vector, and the noisy channel to be characterized by the conditional probability distribution $P_{\mathbf{Y}^{(l)}|\mathbf{Y}^{(l+1)},\mathbf{B}^{(l+1)}}(\mathbf{y}^{(l)}|\mathbf{y}^{(l+1)}, \mathbf{b}^{(l+1)})$,

the signal model for such a channel can be written as follows,

$$\mathbf{Y}^{(l)} = \mathbf{A_B}^{(l,l+1)}\mathbf{Y}^{(l+1)} + \mathbf{Z}^{(l+1)}, \quad l \in [0, L-1] \tag{3.2}$$

where $\mathbf{Z}^{(l+1)} \sim N(0, \Sigma_{\mathbf{z}^{(l+1)}})$ is the additive Gaussian noise vector with independent elements, each corresponding to a different edge from the input layer $l+1$ to the output layer l. Also, $\mathbf{A_B}^{(l,l+1)}$ is the $|\mathbf{Y}^{(l)}| \times |\mathbf{Y}^{(l+1)}|$ sparse connection matrix that also carries the sign information vectors $\mathbf{B}^{(l+1)}$ and $\mathbf{B}^{(l+1)}$. The sparsity of the transition matrix $\mathbf{A_B}^{(l,l+1)} = [\alpha_{ij}]$ is due to assumed underlying tree structure, and it follows the following form

$$\alpha_{ij} = \begin{cases} \gamma_{ij} b_i^{(l)} b_j^{(l+1)} & e_{ij} \in E \\ 0 & e_{ij} \notin E \end{cases} \tag{3.3}$$

where e_{ij} denotes the edge between $Y_i^{(l)}$ and $Y_j^{(l+1)}$. The existence of such edge can be verified from the set E, which is obtained during the learning process. Also, γ_{ij} is the edge-weight showing the correlation value ρ_{ij} up to a sign. Note that, the case for $l = 0$ is a special case, where $\mathbf{A_B}^{(0,1)}$ only depends on $\mathbf{B}^{(1)}$ since there is no sign singularity at the observable layer.

The outputs $\mathbf{Y}^{(l)}$ at each layer l, are generated using the inputs $\mathbf{Y}^{(l+1)}$ at the upper layer. As we will see next, such modeling will be the basis for our successive synthesis scheme. In fact, by starting from the top layer inputs L, at each step we generate the outputs at the lower layer, this will be done till we reach the observed layer to synthesize the Gaussian vector $\mathbf{X}$. Finally, note that in order to take all possible latent tree structures, we need to revise the ordering of layers in certain situations, which will be taken care of in the following subsections. For now, the basic definition for layers will be satisfactory.

3.2.2 Synthesis Approach Formulation

Here, we provide mathematical formulations to address the following fundamental problem: using channel inputs $\mathbf{Y}^{(l+1)}$ and $\mathbf{B}^{(l+1)}$, what are the rate conditions under which we can synthesize the Gaussian channel output $\mathbf{Y}^{(l)}$ with a given $p_{\mathbf{Y}^{(l)}|\mathbf{B}^{(l)}}$. The synthesis channel at each layer is

characterized by (3.2), where the random sequences at any lower layer are affine transformations of their corresponding random upper layer sequences. Note that, at first we are only given $p_{\mathbf{X}}$, but using certain tree learning algorithms we can find those jointly Gaussian latent variables $p_{\mathbf{Y}^{(l)}|\mathbf{B}^{(l)}}$ at every level $l \in [1, L]$. In fact, to account for sign ambiguity we have to deal with mixture Gaussian vectors $p_{\mathbf{Y}^{(l)}}$ at each layer l. We propose a successive synthesis scheme on multiple layers that together induce a latent Gaussian tree, as well as the corresponding bounds on achievable rate tuples. The synthesis scheme is efficient because it utilizes the latent Gaussian tree structure to synthesize the output at each layer. In particular, without resorting to such learned structure we need to characterize $\mathcal{O}(kn)$ parameters (one for each link between a latent and output node) in total, while by considering the sparsity reflected in a tree and each of transition matrices $\mathbf{A_B}^{(l,l+1)}$ we only need to consider $\mathcal{O}(k+n-1)$ parameters (the edges of a tree).

Suppose we *transmit* input messages through N channel uses, in which $t \in \{1, 2, ..., N\}$ denotes the time index. *Transmitting* a random sequence at each layer is equivalent to compute its mapping (the output sequence) through a synthesis channel defined in (3.2). We define $\vec{Y}_t^{(l)}[i]$ to be the t-th symbol of the i-th codeword, with $i \in C_{\mathbf{Y}^{(l)}} = \{1, 2, ..., M_{Y^{(l)}}\}$ where $M_{Y^{(l)}} = 2^{NR_{Y^{(l)}}}$ is the codebook cardinality, transmitted from the existing k_l sources at layer l. We assume there are k_l sources $Y_j^{(l)}$ present at the l-th layer, and the channel has L layers. We can similarly define $\vec{B}_t^{(l)}[k]$ to be the t-th symbol of the k-th codeword, with $k \in C_{\mathbf{B}^{(l)}} = \{1, 2, ..., M_{B^{(l)}}\}$ where $M_{B^{(l)}} = 2^{NR_{B^{(l)}}}$ is the codebook cardinality, regarding the sign variables at layer l. We will further explain that although we define codewords for the Bernoulli sign vectors as well, they are not in fact transmitted through the channel, and rather act as *noisy sources* to select a particular sign setting for latent vector distributions. For *sufficiently* large rates $R_{\mathbf{Y}} = [R_{Y^{(1)}}, R_{Y^{(2)}}, ..., R_{Y^{(L)}}]$ and $R_{\mathbf{B}} = [R_{B^{(1)}}, R_{B^{(2)}}, ..., R_{B^{(L)}}]$ and as N grows the synthesized density of latent Gaussian tree converges to $p_{\mathbf{W}^N(\mathbf{w}^N)}$, i.e., N i.i.d realization of the given output density $p_{\mathbf{W}}(\mathbf{w})$, where $W = \{\mathbf{X}, \mathbf{Y}, \mathbf{B}\}$ is a compound random variable consisting the output, latent, and sign variables. In other words, the average total variation between the two joint densities vanishes as N grows

[16],

$$\lim_{N\to\infty} E||q(\mathbf{w}_1,...,\mathbf{w}_N) - \prod_{t=1}^{N} p_{\mathbf{w}_t}(\mathbf{w}_t)||_{TV} \to 0 \tag{3.4}$$

where $q(\mathbf{w}_1,...,\mathbf{w}_N)$ is the synthesized density of latent Gaussian tree, and $E||.||_{TV}$, represents the average total variation. In this situation, we say that the rates $(R_{\mathbf{Y}}, R_{\mathbf{B}})$ are *achievable* [16]. Our achievability proofs heavily relies on *soft covering lemma* shown in [16]. Loosely speaking, the soft covering lemma states that one can synthesize the desired statistics with arbitrary accuracy, if the codebook sizes (or equivalently, the rates $(R_{\mathbf{Y}}, R_{\mathbf{B}})$) are sufficient and the channel through which these codewords are sent is noisy enough. This way, one can cover the desired statistics up to arbitrary accuracy. The main objective is to maximize such rate region, and develop a proper synthesis scheme to achieve the desired statistics.

For simplicity of notation, we drop the symbol index and use $Y_t^{(l)}$ and $B_t^{(l)}$ instead of $\vec{Y}_t^{(l)}[i]$ and $\vec{B}_t^{(l)}[k]$, respectively, since they can be understood from the context.

3.3 Mutual Information of Layered Synthesis Channels with Correlation Sign Singularity

3.3.1 Properties of Sign Information Vector $\mathbf{B}$

In Theorem 1, whose proof can be found in Appendix 3.6.1, we characterize the size and dependency relations of sign vectors for any general minimal latent Gaussian tree.

Theorem 1. (1) *The correlation values ρ_{yx_i} in regard to the outputs X_i that are connected to a single input, say Y, share an equivalent sign class, i.e., they either all belong to $B = b$ or $B = -b$.*

(2) *Given the cardinality of input vector $\mathbf{Y} = \{Y_1, Y_2, ..., Y_k\}$ is k, then there are totally 2^k minimal Gaussian trees with isomorphic structures, but with different correlation signs that induce the same joint density of the outputs, i.e., equal $p_{\mathbf{X}}(\mathbf{x})$.*

For example, in a Gaussian tree shown in Figure 3.1, there is only one hidden node $Y^{(1)}$, and we already know by previous discussions that there are two latent Gaussian trees with different sign values for $B^{(1)}$, which induce the same output joint density $p_{\mathbf{X}}(\mathbf{x})$. In more general cases

the problem of assigning correlation sign variables is more subtle, where we clarify the approach using two examples, next.

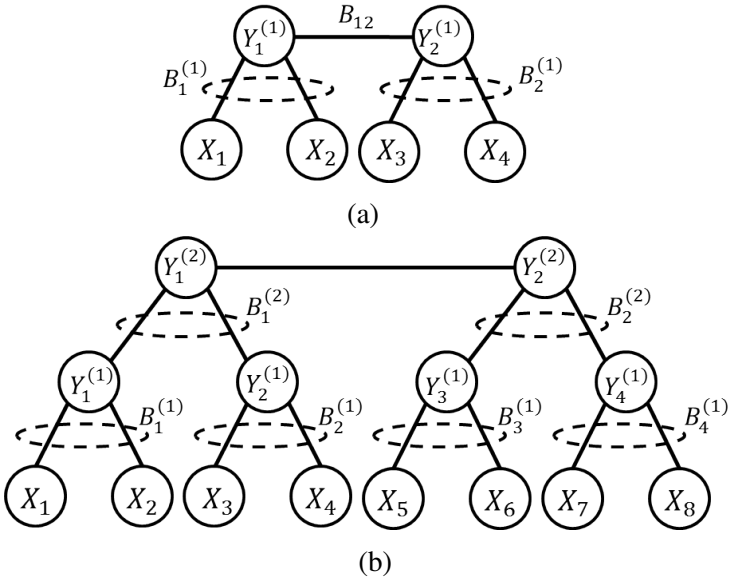


FIGURE 3.2: Two possible cases to demonstrate the dependency relations of sign variables: (a) with two hidden inputs, and (b) with 4 hidden inputs at two layers

In a Gaussian tree shown in Figure 3.2(a) there are two hidden nodes Y_1 and Y_2. By Theorem 1, we know that there are 4 Gaussian trees with sign ambiguity. Also, from the first part in Theorem 1 we may introduce $B_1^{(1)}$ to capture the correlation signs $\rho_{x_1y_1}$ and $\rho_{x_2y_1}$, and $B_2^{(1)}$ for the correlation signs $\rho_{x_3y_2}$ and $\rho_{x_4y_2}$. We can think of B_{12} as the sign of $\rho_{y_1y_2}$. Note that the link between the variables Y_1 and Y_2 are in both groups with common correlation sign, so we anticipate that B_{12} should be dependent on both $B_1^{(1)}$ and $B_2^{(1)}$. Since we need to maintain the correlation signs regarding $\rho_{x_ix_j},\ i \in \{1,2\},\ j \in \{3,4\}$, hence the product $B_1^{(1)}B_2^{(1)}B_{12}$ should maintain its sign. Thus, we have $B_{12} = B_1^{(1)}B_2^{(1)}$, so B_{12} is completely determined given $B_1^{(1)}$ and $B_2^{(1)}$. In other words, we may write the pairwise correlation as $E[Y_1^{(1)}Y_2^{(1)}] = \gamma_{12}B_1^{(1)}B_2^{(1)}$, which further justifies the Gaussian mixture assumption for latent variables $Y_1^{(1)}$ and $Y_1^{(2)}$. Next, consider the Gaussian tree shown in Figure 3.2(b), in which there are six hidden inputs. Similar to the previous case, we introduce four sign variables to capture the first layer sign information. In this case, we

further need to introduce $B_1^{(2)}$ and $B_2^{(2)}$ corresponding to second layer latent inputs. Similar to the previous cases, the pairwise correlation sign of latent inputs depends on the corresponding sign variables at the same layer. For example, $E[Y_1^{(1)}Y_4^{(1)}] = \gamma_{14}B_1^{(1)}B_4^{(1)}$ or $E[Y_1^{(2)}Y_2^{(2)}] = \gamma_{12}B_1^{(2)}B_2^{(2)}$.

One may note the pattern on such definition: The pairwise correlation signs of latent inputs only depends on the corresponding sign variables at the same layer, and it is independent of sign variables at other layers. As we will see shortly, such property is essential in our layered synthesis scheme to make sure the conditional independence of different layers from each other, given the information of neighboring layers.

3.3.2 Single Layer case: Mutual Information between Observables and Latent Variables

It is best to start the achievability discussion by a simple scenario we considered in [53]. In [53] we were only concerned about the synthesis of output vector statistics with given $p_{\mathbf{X}}$. Let us define $\tilde{\mathbf{Y}} = \{\mathbf{Y}, \mathbf{B}\}$, then the formalized problem has the following form:

$$\begin{aligned} &\inf_{p_{\tilde{\mathbf{Y}}}(\tilde{\mathbf{y}})} I(\mathbf{X}; \tilde{\mathbf{Y}}),\ s.t., \\ &\quad p_{\mathbf{X},\tilde{\mathbf{Y}}}(\mathbf{x}, \tilde{\mathbf{y}})\ induces\ a\ minimal\ Gaussian\ tree \\ &\quad X_i \perp X_j | \tilde{\mathbf{Y}} \\ &\quad \Sigma_{\tilde{y}\in\tilde{\mathbf{Y}}} p(\mathbf{x}, \tilde{\mathbf{y}}) = p_{\mathbf{X}}(\mathbf{x}) \end{aligned} \tag{3.5}$$

Remark 2. Due to Markov property, we know that given $\tilde{Y}^{(1)}$, the output layer $\mathbf{X}$ is conditionally independent of all vectors $\tilde{Y}^{(l)},\ l \in [2, L]$ at upper layers. Hence, we have the equality $I(\mathbf{X}; \tilde{\mathbf{Y}}) = I(\mathbf{X}; \tilde{\mathbf{Y}}^{(1)})$, i.e., to synthesize the output vector statistics, all we need are the common latent inputs $\tilde{Y}^{(1)}$ (and of course the independent additive Gaussian noises and Bernoulli sign variables). As we will see shortly, this is a special case to our layered synthesis strategy, where we only deal with a single layer, and want to synthesize the output statistics.

Remark 3. Note that such optimization problem is defined for those output vectors $\mathbf{X}$, whose covariance matrix $\Sigma_{\mathbf{X}}$ is in the subspace of positive definite matrices that induce a latent Gaussian

tree. As discussed earlier, such subspace can be completely characterized by a systems certain inequalities (or equalities in certain cases) between pairwise covariance elements in $\Sigma_{\mathbf{X}}$ [43]. Hence, all of the mutual information values should be evaluated under a given Gaussian tree $G_T(V, E, W)$. For simplicity we drop this notation in their expressions. Hence, such problem is not the same as the general *Wyner's common information* setting, since in Wyner's scenario, no structural constraint is imposed on latent variables.

The *minimality* assumption on the Gaussian tree structure, indicates that in our case $|\mathbf{X}| \geq 3$, i.e., the number of observed variables should be at least three. In a minimal Gaussian tree we assume all the hidden variables have at least three neighbors [2], which results in ignoring all those singular cases where there can be arbitrarily redundant hidden variables added to the model without changing the observed joint density $p_{\mathbf{X}}(\mathbf{x})$. In this setting, by Theorem 2, whose proof can be found in Appendix 3.6.2, we show that regardless of the underlying Gaussian tree structure, there is no room to minimize $I(\mathbf{X}; \tilde{\mathbf{Y}})$.

Theorem 2. *Given $p_{\mathbf{X}}(x) \sim N(0, \Sigma_{\mathbf{x}})$ and the settings in* (4.2)*, the mutual information $I(\mathbf{X}; \tilde{\mathbf{Y}})$ is only a function of $\Sigma_{\mathbf{x}}$ and if the observable nodes are only leaf nodes, the mutual information is given by,*

$$I(\mathbf{X}; \tilde{\mathbf{Y}}) = \frac{1}{2} \log \frac{|\Sigma_{\mathbf{x}}|}{\prod_{i=1}^{n} (1 - \frac{\rho_{x_i x_{j_i}} \rho_{x_i x_{k_i}}}{\rho_{x_{j_i} x_{k_i}}})} \tag{3.6}$$

where for each X_i, we choose two other nodes X_{j_i}, X_{k_i}, where all three of them are connected to each other through Y_{X_i} (i.e., one of their common ancestors), which is one of the hidden variables adjacent to X_i.

Intuitively, given Σ_x and any three outputs that have a common latent variable as their input, the correlation values between each output and the input is fixed, since varying one correlation results in varying the other correlations in the same direction, hence making the pairwise correlation between the other outputs change, which is impossible.

Remark 4. Theorem 2 indicates a special behavior of the mutual information under latent Gaussian tree assumption. In particular, given X_i and its latent parent Y_{X_i} we may end up with several options for X_{j_i} and X_{k_i}. However, it can be shown that in a subspace of correlations corresponding to latent Gaussian trees [43], all those distinct options result in a same value for the term $\rho_{x_i x_{j_i}} \rho_{x_i x_{k_i}} / \rho_{x_{j_i} x_{k_i}}$. In fact, we show that such terms are all equal to $\rho^2_{x_i y_{x_i}} \in (0, 1)$, for $X_i \in \mathbf{X}$ and $Y_{X_i} \in \mathbf{Y}^{(1)}$. In other words, they characterize the correlation between each of the outputs X_i with its corresponding parent Y_{X_i} at the first layer.

Remark 5. Due to equality $I(\mathbf{X}; \mathbf{Y}, \mathbf{B}) = I(\mathbf{X}; \mathbf{Y}^{(1)}, \mathbf{B}^{(1)})$ we can show that to compute the mutual information value in Theorem 2, we only need the correlation values of them form $\rho^2_{x_i y_{pa_i}}$ that are between the observables and their immediate parents. As we will see shortly, this argument can be easily generalized to a multi-layer case, in which to compute the mutual information between the outputs of each layer and the higher layer variables, we only need those inputs that are the parents of output variables, i.e., the variables in a single layer above the outputs.

Note that from (3.6) we can see that the mutual information $I(\mathbf{X}; \mathbf{Y}, \mathbf{B})$ does not depend on sign information, which further justifies our point on intrinsic sign ambiguity in latent Gaussian trees. One may easily deduce the following,

$$I(\mathbf{X}; \tilde{\mathbf{Y}}) = I(\mathbf{X}; \mathbf{Y}, \mathbf{B}) = I(\mathbf{X}; \mathbf{Y}) + I(\mathbf{X}; \mathbf{B}|\mathbf{Y}) \tag{3.7}$$

The results in Theorem 2 combined with (3.7), suggests that by minimizing $I(\mathbf{X}; \mathbf{Y})$, one may eventually maximize $I(\mathbf{X}; \mathbf{B}|\mathbf{Y})$, i.e., quantifying the maximum amount of information loss on the sign input $\mathbf{B}$. In other words, to reach lower synthesis rates and maximizing the achievable rate region, we need to maximize the information loss on sign information. In Theorem 3, whose proof can be found in Appendix 3.6.3 we show that in order to minimize the mutual information $I(\mathbf{X}; \mathbf{Y})$ the sign inputs should be uniformly distributed.

Theorem 3. *Given the Gaussian vector* $\mathbf{X}$ *with* $\Sigma_{\mathbf{x}}$ *inducing a latent Gaussian tree, with latent parameters* $\mathbf{Y}$ *and sign vector* $\mathbf{B}$ *the optimal solution for* $\pi^* = arg\min_{\pi\in[0,1]^k} I(\mathbf{X};\mathbf{Y})$ *happens for uniform sign vector.*

In other words, for all Bernoulli variables $B_i \in \mathbf{B}$ for the optimal solution we should have $\pi_i = 1/2$. Proving this result, relies upon showing the convexity of mutual information $I(\mathbf{X};\mathbf{Y})$ with respect to certain injective functions of π_i. Then, we show that the minimum happens for the case where all such functions are equal, and by converting these values back to π-space we have the desired results.

3.3.3 Multi-layer case: Mutual Information between Outputs and Inputs

Again, considering the successive synthesis perspective, we are interested in generating the output vector $\mathbf{Y}^{(l)}$, using its upper layer inputs $\mathbf{Y}^{(l+1)}$ with minimum amount of necessary random bits. However, note that as shown in previous examples, in general $\mathbf{Y}^{(l)}$ follows a mixture Gaussian model, since its covariance matrix is dependent to the sign vector $\mathbf{B}^{(l)}$. Hence, in order to follow the same principles as in (4.2) the adopted objective function is $\inf_{p_{\mathbf{Y}^{(l+1)}}} I(\tilde{\mathbf{Y}}^{(l+1)};\mathbf{Y}^{(l)}|\mathbf{B}^{(l)})$, where conditioning on each realization of $\mathbf{B}^{(l)}$ results in a Gaussian density for the output vector $\mathbf{Y}^{(l)}$. This is further discussed in detail, when we explain our successive synthesis method in the next subsection. One may wonder whether the conditional independence and minimality constraints in (4.2) also hold in this case. The way we defined each input-output relation in (3.2), we can use similar arguments as before to show the independence of each output vector $\mathbf{Y}^{(l)}|\mathbf{B}^{(l)}$ with the sign input vector $\mathbf{B}^{(l+1)}$, since regardless of the sign input values the conditional output vectors remain jointly Gaussian. Also, by the results of Theorem 2 we know that the overall mutual information $I(\mathbf{X};\tilde{\mathbf{Y}})$ is only a function of observed covariance matrix $\Sigma_{\mathbf{x}}$. So we may conclude that all the pairwise correlations in between any two consecutive layer are fixed, given $\Sigma_{\mathbf{x}}$. Intuitively, such correlations are deduced from a latent tree, whose edge-weights are already determined via $\Sigma_{\mathbf{x}}$ (up to sign). Hence, given $p_{\mathbf{X}}(x) \sim N(0,\Sigma_{\mathbf{x}})$ and assuming $p_{\mathbf{X}\tilde{\mathbf{Y}}}$ induces

a minimal latent Gaussian tree, the input-output mutual information at layers $l+1$ and l, i.e., $I(\tilde{\mathbf{Y}}^{(l+1)};\mathbf{Y}^{(l)}|\mathbf{B}^{(l)})$ for $l \in [0, L-1]$ is already determined by $\Sigma_{\mathbf{x}}$.

3.4 Achievable Rate Regions for Successive Synthesis of Latent Gaussian Tree

In what follows we provide the achievable rate regions to synthesize the Gaussian tree statistics $p_{\mathbf{XY}}$ for three distinct cases that together cover all possible varieties that may happen in latent Gaussian tree structures. As we see, such intuitive classification of Gaussian trees results in better understanding the synthesis scheme for each category.

3.4.1 A Basic Case Study

In this case, we assume that the nodes at each layer are only connected to the nodes at upper/lower layers. In other words, there is no edge between the nodes at the same layer, and they are connected to each other through one or several nodes at the upper layers. Moreover, by deleting all the nodes at the lower layer, all the nodes at current layer should become leaves. To better clarify our approach, it is best to begin the synthesis discussion by several illustrative examples.

Example 3. Consider a latent star topology with Gaussian source $Y^{(1)}$ and sign input $B^{(1)}$, with corresponding output vector $\mathbf{X} = [X_1, X_2, ..., X_n]$. This can be modeled as

$$\begin{bmatrix} X_{1,t} \\ X_{2,t} \\ \vdots \\ X_{n,t} \end{bmatrix} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} B_t^{(1)} Y_t^{(1)} + \begin{bmatrix} Z_{1,t} \\ Z_{2,t} \\ \vdots \\ Z_{n,t} \end{bmatrix}, \; t \in \{1, 2, ..., N\} \tag{3.8}$$

A special case for such broadcast channel is shown in Figure 3.1, where the channel has only three outputs X_1, X_2, and X_3. In the following Corollary we provide the achievable rate region for the broadcast channel As we will show later, this is a special case in Theorem 4, which is due to soft covering lemma and the results in [16].

Corollary 3. *For the latent star topology characterized by* (3.8)*, the following rates are achievable,*

$$R_{Y^{(1)}} + R_{B^{(1)}} \geq I(\mathbf{X}; Y^{(1)}, B^{(1)})$$
$$R_{Y^{(1)}} \geq I(\mathbf{X}; Y^{(1)}) \tag{3.9}$$

Note that the sum of the rates $R_{Y^{(1)}} + R_{B^{(1)}}$ is lower bounded by $I(\mathbf{X}; Y^{(1)}, B^{(1)})$, which by Theorem 2 is fixed. However, the minimum rate for $R_{Y^{(1)}}$ is achieved by $min_{p_{Y^{(1)}}} I(\mathbf{X}; Y^{(1)})$. Also due to Theorem 3 we know that the optimal solution occurs when $B^{(1)}$ is uniformly distributed, i.e., $\pi_1 = 1/2$.

In the synthesis scheme we first need to generate the proper codebook that satisfies the rate conditions in Corollary 3. We generate $2^{NR_{B^{(1)}}}$ codewords to form the codebook $C_{B^{(1)}}$ with proper size for sign variables. Similarly, we generate $2^{NR_{Y^{(1)}}}$ Gaussian codewords to form the codebook $C_{B^{(1)}}$. Note that in general the latent variables have mixture Gaussian distributions, hence, such star tree is a very special case with only one *Gaussian* latent variable. Now, to obtain a Gaussian output sequence, each time we randomly pick codewords $(y^{(1)})^N$ and $(b^{(1)})^N$ from $C_{Y^{(1)}}$ and $C_{B^{(1)}}$, respectively. Based on the observed sign instances $b_t^{(1)},\ t \in [1, ..., N]$ at each time slot, we decide which channel $P_{\mathbf{X}_t|y_t^{(1)}b_t^{(1)}}$ is used to send each $y_t^{(1)}$ to generate the output X_i^N. Figure 3.3 shows the synthesis scheme for this case.

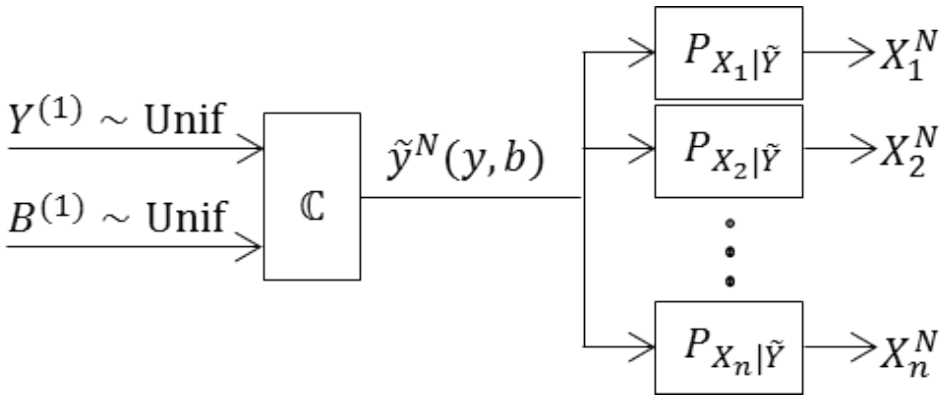


FIGURE 3.3: The synthesis scheme for a latent Gaussian star tree

We may compute the synthesized output as follows,

$$q(\mathbf{x}_1, ..., \mathbf{x}_N) = \frac{1}{M_{\mathbf{B}^{(1)}}} \frac{1}{M_{\mathbf{Y}^{(1)}}} \sum_{i=1}^{M_{\mathbf{Y}^{(1)}}} \sum_{k=1}^{M_{\mathbf{B}^{(1)}}} \prod_{t=1}^{N} p_{\mathbf{X}|\mathbf{Y},\mathbf{B}}(\mathbf{x}_t|\mathbf{y}_t^{(1)}[i]\mathbf{b}_t^{(1)}[k]) \tag{3.10}$$

where the distribution $p_{\mathbf{X}|\mathbf{Y},\mathbf{B}}(\mathbf{x}_t|\mathbf{y}_t^{(1)}[i]\mathbf{b}_t^{(1)}[k])$ represents each channel use t for corresponding input messages, and can be computed via signal model in (3.2).

To synthesize the overall joint distribution, we need to consider the corresponding input codewords $(y^{(1)})^N$ and $(b^{(1)})^N$. In particular, each synthesized output vector $\mathbf{X}^N$ has its own associated input codeword, and we need to pair both of these codewords to achieve the synthesized statistics. Note that such *pairing* strategy is essential, since otherwise the synthesized statistics will not be arbitrarily close to the desired distribution. This will be further discussed in the following examples, where the correspondence between the codewords at each layer should be maintained.

Example 4. Consider the channel shown in Figure 3.2(a). In this case, we are given two hidden inputs $Y_1^{(1)}$ and $Y_2^{(1)}$, and by previous arguments we know $\mathbf{B}^{(1)} = \{B_1^{(1)}, B_2^{(1)}, B_{12}\}$ with $B_{12} = B_1^{(1)} B_2^{(1)}$, completely determined by independent sign variables $B_1^{(1)}$ and $B_2^{(1)}$. We may write,

$$\begin{bmatrix} X_{1,t} \\ X_{2,t} \\ X_{3,t} \\ X_{4,t} \end{bmatrix} = \begin{bmatrix} \alpha_{11} B_{1,t}^{(1)} & 0 \\ \alpha_{21} B_{1,t}^{(1)} & 0 \\ 0 & \alpha_{32} B_{2,t}^{(1)} \\ 0 & \alpha_{42} B_{2,t}^{(1)} \end{bmatrix} \begin{bmatrix} Y_{1,t}^{(1)} \\ Y_{2,t}^{(1)} \end{bmatrix} + \begin{bmatrix} Z_{1,t} \\ Z_{2,t} \\ Z_{3,t} \\ Z_{4,t} \end{bmatrix} \tag{3.11}$$

where $t \in \{1, 2, ..., N\}$ denotes each channel use. Here, two inputs $Y_1^{(1)}$ and $Y_2^{(1)}$ are dependent and their pairwise correlation can be computed via $E[Y_1^{(1)} Y_2^{(1)}] = \gamma_{12} B_{12} = \gamma_{12} B_1^{(1)} B_2^{(1)}$, in which γ_{12} determines the degree of correlation and is learned by certain inference algorithms, e.g., RG or CLRG [2]. Note that the dependency relation of symbols $Y_{1,t}^{(1)}$ and $Y_{2,t}^{(1)}$ follows a Gaussian mixture model, since their covariance is a function of binary inputs $B_{1,t}^{(1)}$ and $B_{2,t}^{(1)}$. But, note that in a given codebook consisting of $M_{\mathbf{Y}^{(1)}}$ codewords, for each realization of $\mathbf{b}_{1,t}^{(1)} \mathbf{b}_{2,t}^{(1)}$

the joint density of $\mathbf{Y}_t^{(1)}$ is Gaussian. Hence, one may divide the codebook $\mathbb{C}$ into two parts $\mathbb{S}_i,\ i \in \{1,2\}$, in which each part follows a specific Gaussian density with covariance values $E[Y_{1,t}^{(1)}Y_{2,t}^{(1)}] = \gamma_{12}b_{1,t}^{(1)}b_{2,t}^{(1)}$. In particular, to generate a codeword we first generate the codebooks $C_{\mathbf{B}^{(1)}}$ and $C_{\mathbf{Y}^{(1)}}$. Note that the generated codewords $(\mathbf{Y}^{(1)})^N \in C_{\mathbf{Y}^{(1)}}$ are mixture Gaussians even at each time slot t. To be precise, at each time slot t, we generate two random Gaussian sample vectors, one with $E[Y_{1,t}^{(1)}Y_{2,t}^{(1)}] = \gamma_{12}$ and the other with $E[Y_{1,t}^{(1)}Y_{2,t}^{(1)}] = -\gamma_{12}$. Then, similar to the previous example, at synthesis step and based on the picked sign codeword, we decide which of the two sample vectors should be chosen. The achievable region can be obtained from (3.9), and by replacing $Y^{(1)}$ with $\{Y_1^{(1)}, Y_2^{(1)}\}$ and $B^{(1)}$ with $\{B_1^{(1)}, B_2^{(1)}\}$. Similarly, by Theorem 3 we may conclude that the optimal solution (π_1^*, π_2^*) to $arg\min_{\pi_1,\pi_2} I(\mathbf{X};\mathbf{Y})$ is at $(1/2, 1/2)$.

Let us address more general cases, where we are having a multi-layered latent Gaussian tree with no edge between the variables at the same layer. In other words, the variables at each layer are conditionally independent of each other given the variables at their upper layer. Moreover, by deleting all the nodes at the lower layer, all the nodes at current layer should become leaves. This, in turn forms a *hyper-chain* structure for latent Gaussian tree, where the *hyper-nodes* consist of every variable at the same layer, and *hyper-edges* are the collection of links connecting each the nodes at each adjacent layer. Figure 3.4 shows the general synthesis scheme. At each layer i, we define $\tilde{\mathbf{Y}}^{(i)} = \{\mathbf{Y}^{(i)}, \mathbf{B}^{(i)}\}$ to be the combination of input vectors. This situation is a little more subtle than the previous single-layered cases, since we need to be more cautious on specifying the rate regions as well as the synthesis scheme.

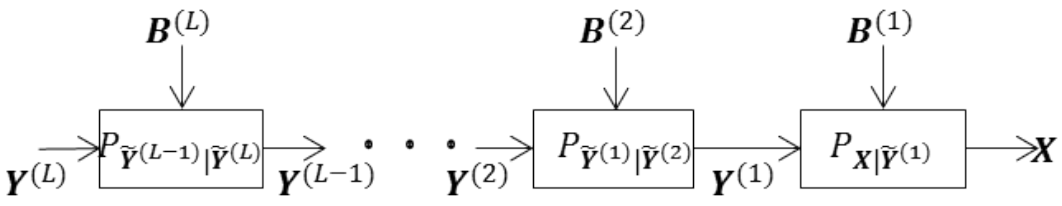


FIGURE 3.4: Multi-layered output synthesis

Example 5. To clarify, consider the case shown in Figure 3.2(b), in which the Gaussian tree has two layers of inputs. Similar as previous cases we can compute the pairwise covariance between inputs at the first layer as $E[Y_{k,t}^{(1)}Y_{l,t}^{(1)}] = \gamma_{kl}B_{k,t}^{(1)}B_{l,t}^{(1)}$, in which $k \neq l \in \{1,2,3,4\}$. By the previous example, we know that the input vector $\mathbf{Y}_t^{(1)}$ is Gaussian for each realization of $\mathbf{B}_t^{(1)} = \{\mathbf{b}_{1,t}^{(1)}, \mathbf{b}_{2,t}^{(1)}, \mathbf{b}_{3,t}^{(1)}, \mathbf{b}_{4,t}^{(1)}\}$. Hence, one may divide the codebook $\mathbb{C}$ into $2^{k_1} = 16$ parts $\mathbb{S}_i,\ i \in \{1,2,...,16\}$, in which each part follows a specific Gaussian density with covariance values $E[Y_{k,t}^{(1)}Y_{l,t}^{(1)}] = \gamma_{kl}b_{k,t}^{(1)}b_{l,t}^{(1)},\ k \neq l \in \{1,2,3,4\}$. Now, for each subset, at the second layer we are dealing with the case shown in Figure 3.2(a), which has been resolved. Thus, the lower bound on the achievable rates in the second layer are as follows,

$$\begin{aligned} R_{\mathbf{Y}^{(2)}} &\geq I(\mathbf{Y}^{(1)}; \mathbf{Y}^{(2)}|\mathbf{B}^{(1)}) \\ R_{\mathbf{Y}^{(2)}} + R_{\mathbf{B}^{(2)}} &\geq I(\mathbf{Y}^{(1)}; \mathbf{Y}^{(2)}, \mathbf{B}^{(2)}|\mathbf{B}^{(1)}) \end{aligned} \tag{3.12}$$

This is due to the fact that we compute subsets of codebook for each realization of $\mathbf{B}^{(1)}$. Let us elaborate the successive codebook generation scheme in this case.

First, we need to generate the codebooks at each layer, beginning from the top layer all the way to the first layer. The sign codebooks $C_{\mathbf{B}^{(2)}}$ and $C_{\mathbf{B}^{(1)}}$ are generated beforehand, and simply regarding the Bernoulli distributed sign vectors $\mathbf{B}^{(2)}$, and $\mathbf{B}^{(1)}$. Hence, each sign codeword is a sequence of vectors consisting elements chosen from $\{-1, 1\}$. We may also generate the top layer codebook $C_{\mathbf{Y}^{(2)}}$ using mixture Gaussian codewords, where each codeword at each time slot consists of all possible sign realizations of $\mathbf{B}_t^{(2)}$. Each of these settings characterize a particular Gaussian distribution for the top layer latent variables $\mathbf{Y}^{(2)}$. The necessary number of codewords needed is $M_{\mathbf{Y}^{(2)}} = 2^{NR_{\mathbf{Y}^{(2)}}}$, where the rate in the exponent is lower bounded and characterized using (3.12). To form the second codebook $C_{\mathbf{Y}^{(1)}}$, we know that we should use the codewords in $C_{\mathbf{Y}^{(2)}}$. We randomly pick codewords from $C_{\mathbf{Y}^{(2)}}$ and $C_{\mathbf{B}^{(2)}}$. Now, based on the chosen sign codeword, we form the sequence $(\mathbf{y}^{(2)}|\mathbf{b}^{(2)})^N$ to be sent through the channels. The sign vector $\mathbf{B}^{(1)}$ consists of $k_1 = 4$ sign variables, hence, resulting in $2^{k_1-1} = 8$ different channel realizations.

Hence, we pass the chosen sequence through 8 different channels $p_{\mathbf{Y}^{(1)}|\tilde{\mathbf{Y}}^{(2)}\mathbf{b}^{(1)}}$. This way, we send the chosen codeword through the 8 noisy channels to produce a particular codeword in $C_{\mathbf{Y}^{(1)}}$. It is important to note that we showed in subsection 3.3.3 that although each of these channel correspond to different sign realizations of $\mathbf{B}^{(1)}$ vector, however, due to underlying latent Gaussian tree assumption they maintain the same rate. Note that, such produced codeword is in fact a collection of Gaussian vectors, each corresponding to a particular sign realization $\mathbf{b}^{(l)} \in \mathbf{B}^{(1)}$. We iterate this procedure $M_{\mathbf{Y}^{(1)}}$ times to produce enough codewords that are needed for synthesis requirements of the next layer. The necessary size of $M_{\mathbf{Y}^{(1)}}$ is lower bounded by Corollary 3.

Figure 3.5, shows the described synthesis procedure. In order to produce an output sequence, all we need to do is to randomly pick codewords from $C_{\mathbf{Y}^{(1)}}$ and $C_{\mathbf{B}^{(1)}}$. Then, depending on each time slot sign realization $\mathbf{b}_t^{(1)}$ we use the corresponding channel $p_{\mathbf{X}|\mathbf{Y}^{(1)}\mathbf{b}_t^{(1)}}$ to generate a particular output sequence $\mathbf{X}^N$.

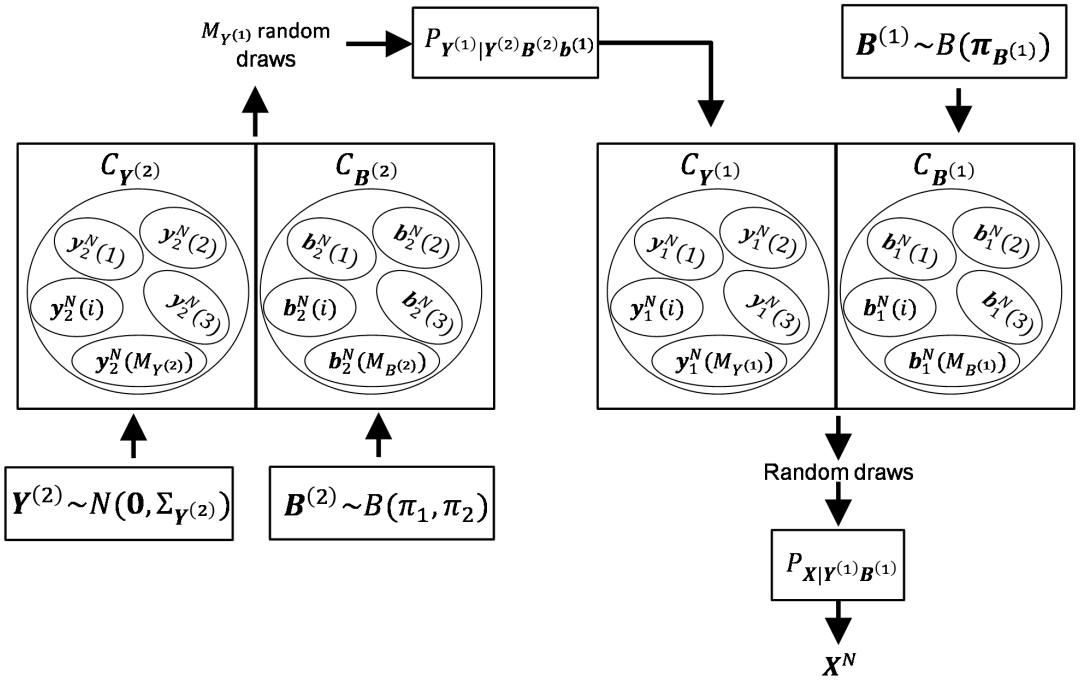


FIGURE 3.5: The proposed codebook generation scheme used for a Gaussian tree shown in 3.2(b). The codebook size $|C_{\mathbf{Y}^{(2)}}|$ at the top layer with shown corresponding distribution is determined by the input-output mutual information in the channel $p_{\mathbf{Y}^{(1)}|\mathbf{Y}^{(2)}\mathbf{B}^{(2)},\mathbf{b}^{(1)}}$. To obtain $C_{\mathbf{Y}^{(1)}}$, we randomly pick codewords from $C_{\mathbf{Y}^{(2)}}$ and $C_{\mathbf{B}^{(2)}}$ to construct $(\mathbf{Y}^{(2)}|\mathbf{B}^{(2)})^N$. Then, we send it through the eight channels $p_{\mathbf{Y}^{(1)}|\mathbf{Y}^{(2)}\mathbf{B}^{(2)},\mathbf{b}^{(1)}}$ to obtain a particular codeword $(\mathbf{Y}^{(1)})^N$.

In general, the output at the l-th layer $\mathbf{Y}^{(l)}$ is synthesized by $\mathbf{Y}^{(l+1)}$ and $\mathbf{B}^{(l+1)}$, which are at layer $l+1$, and through different channel realizations through $\mathbf{B}^{(l)}$. Algorithm 2 shows the general codebook generation procedure for a Gaussian tree with L layers.

Input: The needed codebook sizes $M_{\mathbf{Y}^{(l)}}$ and $M_{\mathbf{B}^{(l)}}$ for $l \in [1, L]$
Output: The generated codebooks for each layer l
for $l := L$ *to* 1 **do**
  for $i := 1$ *to* $M_{\mathbf{B}^{(l)}}$ **do**
    Randomly generate sign codewords $(\mathbf{b}^{(l)})^N$ to form $C_{\mathbf{B}^{(l)}}$;
  end
end
for $i := 1$ *to* $M_{\mathbf{Y}^{(L)}}$ **do**
  for $\mathbf{b}^{(L)} \in \mathbf{B}^{(L)}$ **do**
    Randomly generate sign codewords $(\mathbf{y}^{(L)})^N|\mathbf{b}^{(L)}$;
  end
  The codewords $(\mathbf{y}^{(L)})^N = \cup_{\mathbf{b}^{(L)}}(\mathbf{y}^{(L)})^N|\mathbf{b}^{(L)}$ form $C_{\mathbf{Y}^{(L)}}$;
end
for $l := L-1$ *to* 1 **do**
  for $i := 1$ *to* $M_{\mathbf{Y}^{(l)}}$ **do**
    Randomly pick a codeword $(\mathbf{y}^{(l+1)})^N$ from $C_{\mathbf{Y}^{(l+1)}}$;
    Randomly pick a codeword $(\mathbf{b}^{(l+1)})^N$ from $C_{\mathbf{B}^{(l+1)}}$;
    Form the combined codeword $(\mathbf{y}^{(l+1)}|\mathbf{b}^{(l+1)})^N$;
    for $\mathbf{b}^{(l)} \in \mathbf{B}^{(l)}$ **do**
      Send the codeword $(\mathbf{y}^{(l+1)}|\mathbf{b}^{(l+1)})^N$ through the channel $P_{\mathbf{Y}^{(l)}|\mathbf{Y}^{(l+1)},\mathbf{B}^{(l+1)},\mathbf{b}^{(l)}}$ to obtain $(\mathbf{y}^{(l)})^N|\mathbf{b}^{(l)}$;
    end
    The codewords $(\mathbf{y}^{(l)})^N = \cup_{\mathbf{b}^{(l)}}(\mathbf{y}^{(l)})^N|\mathbf{b}^{(l)}$ form $C_{\mathbf{Y}^{(l)}}$
  end
end

ALGORITHM 2. Codebook Generation for each layer of latent Gaussian tree with L layers

Therefore, to synthesize the Gaussian tree statistics that is close enough to the true Gaussian tree distribution, we first need to generate the top layer codebook $C_{\mathbf{Y}^{(l)}}$, and the sign codebooks $C_{\mathbf{B}^{(l)}},\ l \in [1, L]$. Note that the independent Gaussian noises are needed in our synthesis scheme as given source of randomness. In Theorem 4, whose proof can be found in Appendix 3.6.4 we obtain the achievable rate region for multi-layered latent Gaussian tree, while taking care of sign

information as well, i.e., at each layer dividing a codebook into appropriate sub-blocks capturing each realization of sign inputs.

Theorem 4. *For a latent Gaussian tree having L layers, and forming a hyper-chain structure, the achievable rate region is characterized by the following inequalities for each layer l,*

$$\begin{aligned} R_{\mathbf{B}^{(l+1)}} + R_{\mathbf{Y}^{(l+1)}} &\geq I[\mathbf{Y}^{(l+1)}, \mathbf{B}^{(l+1)}; \mathbf{Y}^{(l)} | \mathbf{B}^{(l)}] \\ R_{\mathbf{Y}^{(l+1)}} &\geq I[\mathbf{Y}^{(l+1)}; \mathbf{Y}^{(l)} | \mathbf{B}^{(l)}], \ l \in [0, L-1] \end{aligned} \tag{3.13}$$

where $l = 0$ shows the observable layer, in which there is no conditioning needed, since the output vector $\mathbf{X}$ is already assumed to be Gaussian. Notice that using Theorem 2, we can partially characterize the first lower bound on the sum of rates, since this is a fixed quantity, given the observables covariance matrix; however, analytically characterizing the lower bound on each of the rates $R_{\mathbf{Y}^{(l+1)}}$ due to presence of mixture Gaussian inputs $\mathbf{Y}^{(l+1)}$ is a hard problem to solve. We refer the reader to [54] for further results on mixture Gaussian variables.

Let us assume using the top-down approach shown in Algorithm 2 we generate the appropriate codebooks at each layer. To pick appropriate sample codeword, each time we need to keep track of input-output codewords relationship. Considering each particular layer outputs, we keep track of the corresponding input codeword that generated such output. For instance, consider the same Gaussian tree shown in Figure 3.6. To generate an output sequence $\mathbf{x}^N$, we randomly pick two codewords $(\mathbf{y}^{(1)})^N$ and $(\mathbf{b}^{(1)})^N$ from the corresponding codebooks. The sign codeword decides which channel to be used in order to obtain the outputs. Hence, there is a correspondence between such input codewords and the generated outputs. Similarly, the codeword $(\mathbf{y}^{(1)})^N$ is an output of the top layer inputs, generated by randomly chosen codewords $(\mathbf{y}^{(2)})^N$ and $(\mathbf{b}^{(2)})^N$. Figure 3.6 shows the synthesis scheme that is proposed for the two-layered latent Gaussian tree shown in Figure 3.2b.

In Figure 3.6, different colors in codebooks correspond to different sign realizations. For example, as we know the top layer sign inputs $\mathbf{B}^{(2)}$ can have $2^{k_2-1} = 2$ different sign realizations,

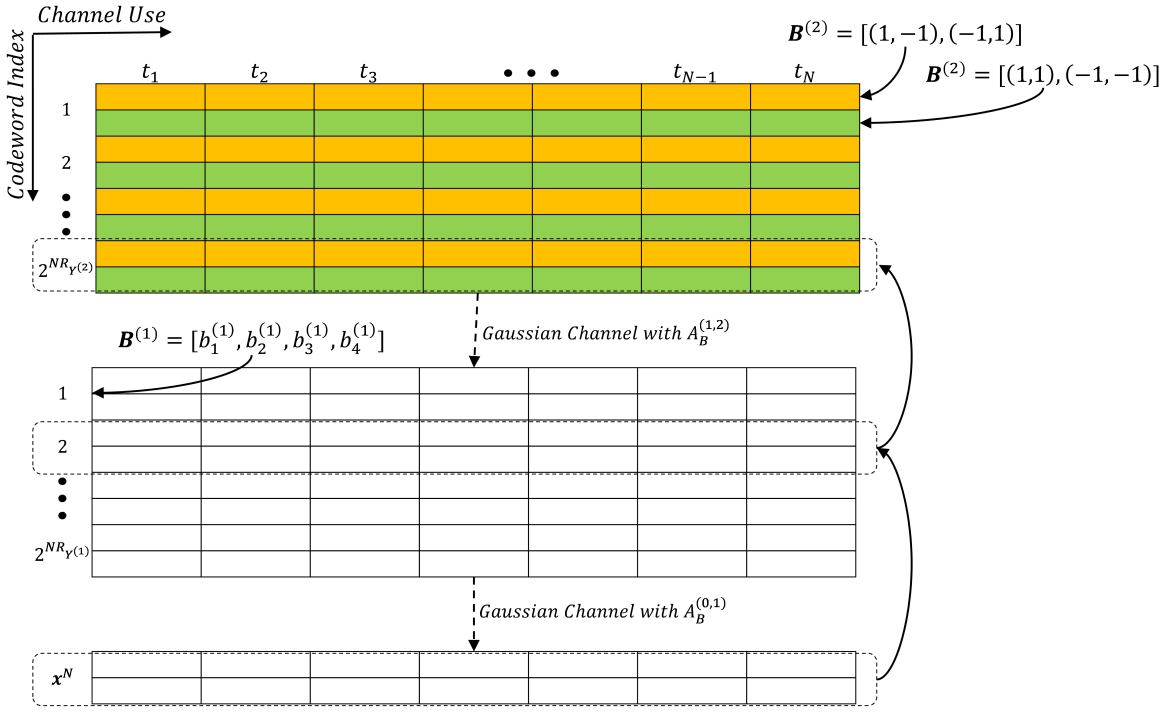


FIGURE 3.6: Synthesis approach for the Gaussian tree in Figure 3.2b

hence 2 different colors are shown in the corresponding codebook. Also, note that each cell in codebooks may contain a vector of samples, due to the fact that each layer usually contains more than one variable. For example, each cell in the top codebook contains two Gaussian samples, corresponding to $\mathbf{y}_t^{(2)}|\mathbf{b}_t^{(2)}$ where $\mathbf{y}^{(2)} = [y_1^{(2)}, y_2^{(2)}]$. The bottom-up synthesis approach first randomly picks the sequences $(\mathbf{y}^{(1)})^N$ and $(\mathbf{b}^{(1)})^N$ from $C_{\mathbf{Y}^{(1)}}$ and $C_{\mathbf{B}^{(1)}}$, respectively, and forms $(\mathbf{y}^{(1)}|\mathbf{b}^{(1)})^N$, then finds the corresponding input codeword $(\mathbf{y}^{(2)}|\mathbf{b}^{(2)})^N$ that generated such codeword at the first layer. Then, the chosen codeword $(\mathbf{y}^{(1)}|\mathbf{b}^{(1)})^N$ is used to generate the output vector $\mathbf{x}^N$ through the given Gaussian channel. The sequence of samples in this case (as shown in Figure 3.6) is $[\mathbf{x}^N, (\mathbf{y}^{(1)}|\mathbf{b}^{(1)})_2^N, (\mathbf{y}^{(2)}|\mathbf{b}^{(2)})_{M_{Y^{(2)}}}^N]$. Remember that each layer's codeword carries its corresponding sign information characterized in codebook generation step.

In general, this procedure should always hold from the bottom to top of latent Gaussian tree, in order to keep a valid joint dependency among the variables at every layer. Algorithm 3 shows this procedure for any general Gaussian tree.

Input: Generated codebooks from Algorithm 2
Output: A valid sequence $(\mathbf{x}^N, (\mathbf{y}|\mathbf{b})^N)$ from the synthesized Gaussian distribution
Randomly pick the codewords $(\mathbf{y}^{(1)})^N$ and $(\mathbf{b}^{(1)})^N$ from $C_{\mathbf{Y}^{(1)}}$ and $C_{\mathbf{B}^{(1)}}$, respectively, and form $(\mathbf{y}^{(1)}|\mathbf{b}^{(1)})^N$;
for $l := 1$ *to* $L-1$ **do**
  For each codeword $(\mathbf{y}^{(l)}|\mathbf{b}^{(l)})^N$, pick the corresponding codeword $(\mathbf{y}^{(l+1)}|\mathbf{b}^{(l+1)})^N$, which has been employed generating it ;
end
For $(\mathbf{y}^{(L-1)}|\mathbf{b}^{(L-1)})^N$, pick the corresponding codeword $(\mathbf{y}^{(L)}|\mathbf{b}^{(L)})^N$ at layer L
Send the chosen codeword $(\mathbf{y}^{(1)}|\mathbf{b}^{(1)})^N$ through the channel $p_{\mathbf{X}|\mathbf{Y}^{(1)}\mathbf{B}^{(1)}}$ to obtain $\mathbf{x}^N$
Output the overall sequence of codewords $[\mathbf{x}^N, (\mathbf{y}^{(1)}|\mathbf{b}^{(1)})^N, ..., (\mathbf{y}^{(L)}|\mathbf{b}^{(L)})^N]$;

ALGORITHM 3. Synthesis approach for a latent Gaussian tree with L layers

3.4.2 The Case with Observables Adjacent with More Than One Latent Variable

Here, we consider more general cases, which may allow nodes at each layer to have more than one neighbor from upper layer. This way, by deleting the nodes at the lower layer, we may end up with several internal (non-leaf) nodes at the current layer. We need to propose a revised achievability proof to characterize the achievable rate region. To clarify our approach, consider the following example shown in Figure 3.7.

Example 6. This is a double layer latent Gaussian tree, with X_6 as an internal node. As it can be seen, after the first step by summing out X_6, we created a clique at the next layer. This is not a latent Gaussian tree structure anymore. In fact, this can be seen as a *junction tree* structure. The problem with such structure is that, given the nodes at upper layer, i.e, $Y_1^{(2)}$, the nodes at the lower layer, i.e, $Y_i^{(1)},\ i \in [2,4]$ are not conditionally independent anymore. This, violates some of the constraints in the acheivability results in Theorem 4.

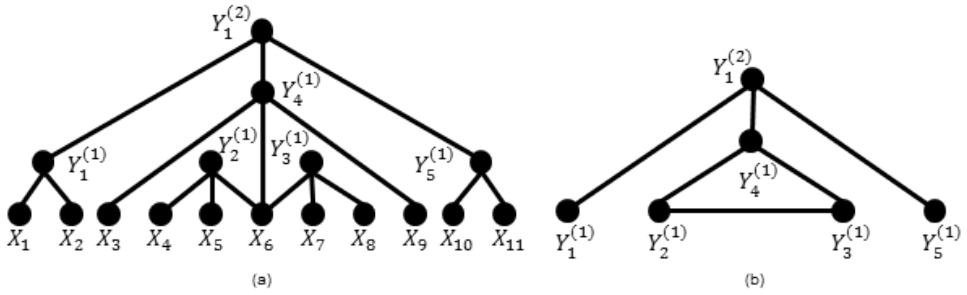


FIGURE 3.7: (a) The original two layered Gaussian tree (b) Obtained grpah after the first iteration

To address such problem, we introduce another *latent pseudo-node*, $Y_2^{(2)}$ and connect it to all the nodes forming the clique, as it is shown in Figure 3.8.

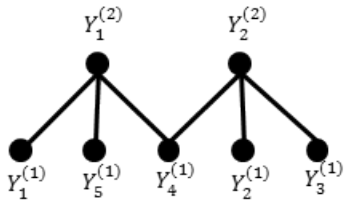


FIGURE 3.8: The intermediate step needed to address the internal node issue: By adding a pseudo node $Y_2^{(2)}$ we break the clique and trun it into a tree structure again

Note that we certainly can represent the formed clique via a latent Gaussian tree (star tree), since from the first iteration we already know that the hidden nodes at the first layer are connected to each other through X_6. So, now after adding the *pseudo-node* $Y_2^{(2)}$, we know that the following equality holds for the new correlations $\rho_{y_2^{(2)}y_i^{(1)}} = \rho_{x_6 y_i^{(1)}},\ i \in [2,4]$. Hence, we may see $Y_2^{(2)}$ as the *mirror* node to X_6, which is added to the nodes in the second layer. Finally, we only need to update the set of nodes at the second layer to $Y^{(2)} = \{Y_1^{(2)}, Y_2^{(2)}\}$.

Remark 6. In general, the synthesis scheme will remain the same as the basic case, with one tweak: at each layer $l+1$ we may need to perform an intermediate step, in which we transform cliques into latent trees (star structure), by adding enough pseudo-nodes to the set of upper layer latent nodes. The sufficient number of pseudo nodes to be added should be the same as the number of internal nodes at layer l. Through such procedure, due to the addition of new nodes (the pseudo nodes) both the corresponding rates $R_{\mathbf{Y}^{(l+1)}}$ and $R_{\mathbf{B}^{(l+1)}}$, and consequently the achievable rate regions will be changed to the following.

$$
\begin{aligned}
R_{\mathbf{B}'^{(l+1)}} + R_{\mathbf{Y}'^{(l+1)}} &\geq I[\mathbf{Y}'^{(l+1)}, \mathbf{B}'^{(l+1)}; \mathbf{Y}^{(l)} | \mathbf{B}^{(l)}] \\
R_{\mathbf{Y}'^{(l+1)}} &\geq I[\mathbf{Y}'^{(l+1)}; \mathbf{Y}^{(l)} | \mathbf{B}^{(l)}]
\end{aligned} \tag{3.14}
$$

where $\mathbf{Y}'^{(l+1)} = \mathbf{Y}^{(l+1)} \cup \mathbf{Y}_p^{(l)}$ and $\mathbf{B}'^{(l+1)} = \mathbf{B}^{(l+1)} \cup \mathbf{B}_p^{(l)}$ are the new input vectors at layer $l+1$, with $\mathbf{Y}_p^{(l)}$ and $\mathbf{B}_p^{(l)}$, being the newly added psuedo latent and sign inputs.

3.4.3 The Case with Observables at Different Layers

Consider a case where an edge is allowed between the variables at the same layer. In this situation we violate a conditional independence constraint used in achievability proof of the basic case, since due to presence of such intra-layer links, given the upper layer inputs, the conditional independence of lower layer outputs is no longer guaranteed. However, again by revising the proof procedure we may show the achievability results in this case as well. To address this issue we need to reform the latent Gaussian tree structure by choosing an appropriate root such that the variables in the newly introduced layers mimic the basic scenario, i.e., having no edges between the variables at the same layer. In particular, we begin with the top layer nodes, and as we move to lower layers we seek each layer for the adjacent nodes at the same layer, and move them to a newly added layer in between the upper and lower layers. In this way, we introduce new layers consisting of those special nodes, but this time we are dealing with a basic case. Note that such procedure might place the output variables at different layers, i.e., all the output variables are not generated using inputs at a single layer. We only need to show that using such procedure and previously define achievable rates, one can still simulate output statistics with vanishing total variation distance. To clarify, consider the following example in Figure 3.9.

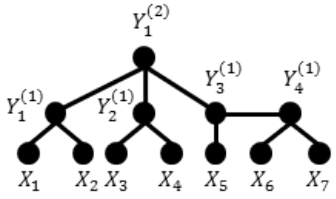


FIGURE 3.9: Latent Gaussian tree with adjacent nodes at layer 1

Example 7. As it can be seen, there are two adjacent nodes in the first layer, i.e., $Y_3^{(1)}$ and $Y_4^{(1)}$ are connected. Using the explained procedure, we may move $Y_4^{(1)}$ to another newly introduced layer, then we relabel the nodes again to capture the layer orderings. The reformed Gaussian tree is shown in Figure 3.10. In the new ordering, the output variables X_6 and X_7 will be synthesized

one step after other inputs. The input $Y_1^{(3)}$ is used to synthesize the outputs vector $\mathbf{Y}^{(2)}$, which such vector used to generate the first layer outputs, i.e., X_1 to X_5 and $Y_1^{(1)}$. At the last step, the input $Y_1^{(1)}$ will be used to simulate the output pair X_6 and X_7. By Theorem 4 we know that both simulated densities regarding to $q_{X_1^N,X_2^N,X_3^N,X_4^N,X_5^N}$ and $q_{X_6^N,X_7^N}$ approach to their corresponding densities as N grows. We need to show that the overall simulated density $q_{\mathbf{X}^N}$ also approaches to $\prod_{t=1}^{N} p_{\mathbf{X}}(\mathbf{x}_t)$ as well.

We need to be particularly cautious in keeping the joint dependency among the generated outputs at different layers: For each pair of outputs (X_6^N, X_7^N), there exists an input codeword $(Y_1^1)^N$, which corresponds to the set of generated codewords $(X_1^N, X_2^N, X_3^N, X_4^N, X_5^N)$, where together with $(Y_1^1)^N$ they are generated using the second layer inputs. Hence, in order to maintain the overall joint dependency of the outputs, we always need to match the correct set of outputs X_1^N to X_5^N to each of the output pairs (X_6^N, X_7^N), where this is done via $(Y_1^1)^N$.

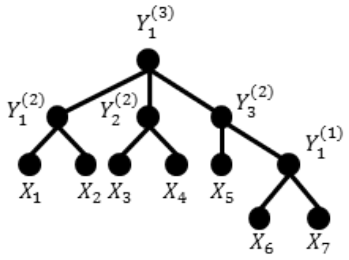


FIGURE 3.10: Another layer introduced to address the issue

In general, we need to keep track of the indices of generated output vectors at each layer and match them with corresponding output vector indices at other layers. This is shown in Lemma 7, whose proof can be found in Appendix 3.6.5,

Lemma 7. *For a latent Gaussian tree having L layers, and not containing an internal node at any iteration, by rearranging each layer so that there is no intra-layer edges, the achievable rate region at each layer l is characterized by the same inequalities as in* (3.13).

By Lemma 7 we may easily extend our results and show that, interestingly, to generate a latent Gaussian tree, we only need its top layer nodes acting as common random sources (and independent Gaussian and Bernoulli noises) to synthesize the entire Gaussian tree structure. Algorithms 2 and 3 are used for the reformed structure for codebook generation and synthesis steps, respectively.

Corollary 4. *Given any latent Gaussian tree consisting of L hidden layers along with an output vector $\mathbf{X}$, by combining the aforementioned procedures described in the last three subsections and using the top layer inputs, i.e., the inputs at the L-th layer, the independent Gaussian noises, and the independent Bernoulli variables, the entire set of nodes in a latent Gaussian tree can be synthesized if the rates at each layer satisfy the constraints captured in* (3.13).

Note that, the top layer nodes, without considering any other node in a tree, will certainly form a chain (or a single node in a special case) structure.

3.5 Conclusion

In this chapter, we formulated a synthesis problem through layered forwarding channels to synthesize those statistics that characterize the latent Gaussian tree structures. Then we deduced an interesting conclusion under which maximizing the achievable rate region also resulted in quantifying the maximum amount of lost information on pairwise correlation signs. Through three different cases we found the achievable rate regions to correctly synthesize the Gaussian outputs, satisfying specific set of constraints. Our layered synthesis approach is shown to be efficient and accurate in terms of reduced required number of parameters needed to synthesize the output statistics, and its closeness to the desired statistics in terms of their total variation distance.

3.6 Proof of Theorems

3.6.1 Proof of Theorem 1

First, let's prove the first part. Consider the case in Figure 3.11. The hidden node y, has k observable neighbors $\{x_1, ..., x_k\}$, while it is connected through two or more edges to other observable

nodes $\{x_{k+1}, ..., x_n\}$. Given only observable covariance matrix Σ_x, we can compute the empirical pairwise covariance values, hence all $\rho_{x_i x_j}$ are fixed.

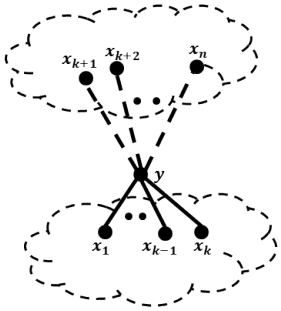


FIGURE 3.11: Neighborhood of hidden variable y

Without loss of generality, suppose we flip the sign of $\rho_{x_1 y}$. To maintain the same covariance matrix Σ_x, the sign of all $\rho_{x_j y}$, $j \in \{2, ..., k\}$ should be flipped. Since, we know $\rho_{x_1 x_j} = \rho_{x_1 y}\rho_{x_j y}$, for all $j \in \{2, ..., k\}$ is fixed. Also, the sign of all pairwise covariance values between y and x_i, for all $i \in \{k+1, ..., n\}$ should be flipped. The same argument as the previous case can be used. However, in this case, all we know is that odd number of sign-flips for the edge-weights between each y and x_i should happen. Using the above arguments, we can see that all $\rho_{x_j y}$, for $j \in \{1, ..., k\}$ maintain their signs, or otherwise all of their signs should be flipped.

For the second part, We inductively show that given a minimal latent tree, with n observable $x_1, ..., x_n$ and with k hidden nodes $y_1, ..., y_k$, we can find 2^k latent trees with different edge-signs that induce the same Σ_x. This is already shown for the star tree shown in Figure 3.1. Suppose such claim holds for all Gaussian trees with $k' < k$ latent nodes. Consider an arbitrary latent tree with k hidden nodes and n observable. Some of these hidden nodes certainly have leaf observable neighbors, which we group them together. Now, note that the problem of finding equivalent sign permutations in this tree can be translated into a problem with smaller tree: Delete all of those leaf observable groups, and treat their hidden parent y_i as their representative. Suppose there are m hidden nodes $\{y_1, ..., y_m\}$, which can represent each of these groups. This case is illustrated in Figure 3.12. Note, as depicted by this Figure, the internal observables as well as those leaf

observables directly connected to them remain intact. By replacing all of these groups with a single node y_i, $i \in \{1, 2, ..., m\}$, we obtain a smaller tree. Now, we can simply assume that all $y_1, ..., y_m$ are observable and their pairwise covariance values are determined. Hence, this tree only has $k - m$ remaining hidden nodes, so due to inductive step it has 2^{k-m} possible equivalent trees with different edge-signs.

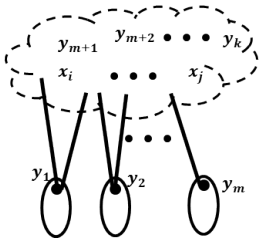


FIGURE 3.12: Figure illustrating the inductive proof

It remains to show that by adding back those m groups of observable, we obtain the claimed result. Add back two groups corresponding to y_1 and y_2. Now, y_1 and y_2 can be regarded as hidden nodes, so now there are $k - m + 2$ hidden nodes, which due to inductive step has 2^{k-m+2} equivalent representations of edge-weights. This can be shown up to $m-1$-th step by adding back the groups for $y_1, ..., y_{m-1}$ nodes, and having a size of $k - 1$ nodes, and again due to induction having 2^{k-1} equivalent sign combinations. By adding back the m-th group, we can obtain two equivalent classes: $b^{(m)}$ or $-b^{(m)}$, where $b^{(m)}$ shows the sign value of the m-th group. This is shown in Figure 3.13 Hence, we obtain $2 \times 2^{k-1} = 2^k$ edge-signs.

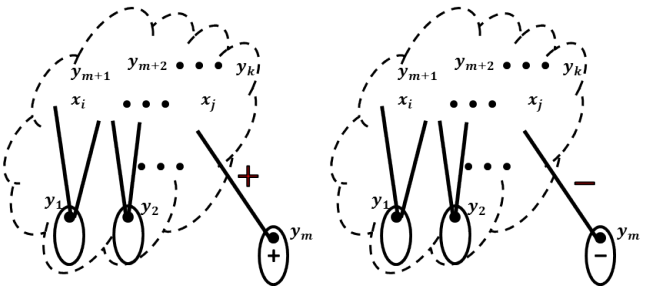


FIGURE 3.13: Obtaining m-th step from $m - 1$-th step
Ãś

This completes the proof.

3.6.2 Proof of Theorem 2

Let's first show that the mutual information $I(\mathbf{X}, \tilde{\mathbf{Y}})$ given $\Sigma_{\mathbf{x}}$ is only a function of pairwise correlations $\rho_{x_i x_j}$, for all $x_i, x_j \in \mathbf{X}$. In a latent Gaussian tree, three cases may happen: The edges can be between two observable, an observable and a latent node, or between two latent nodes.

(1) x_i and x_j are either adjacent or they are connected only through several observables. In this case, since all the pairwise correlations along the path are determined given $\Sigma_{\mathbf{x}}$, so the correlation values are fixed.

(2) x_i and x_j are not adjacent and there is at least one hidden node, e.g., y_1 connecting them. First, suppose y_1 and x_i are adjacent. Since, we assume the tree is minimal, so there should be at least another observable x_k that is connected (but not necessarily adjacent) to y_1. Hence, y_1 acts as a common ancestor to x_i, x_j, and x_k. By changing $\rho_{x_i y_1}$ to another value $\rho'_{x_i y_1}$, by equation $\rho_{x_i x_j} = \rho_{x_i y_1}\rho_{x_j y_1}$ we have to change $\rho_{x_j y_1}$ to $\rho'_{x_j y_1} = \frac{\rho_{x_i y_1}}{\rho'_{x_i y_1}}\rho_{x_j y_1}$. Similarly, by equality $\rho_{x_i x_k} = \rho_{x_i y_1}\rho_{x_k y_1}$, we know $\rho'_{x_k y_1} = \frac{\rho_{x_i y_1}}{\rho'_{x_i y_1}}\rho_{x_k y_1}$. However, by another equality $\rho_{x_j x_k} = \rho_{x_j y_1}\rho_{x_k y_1}$, we deduce $\rho'_{x_k y_1} = \frac{\rho_{x_j y_1}}{\rho'_{x_j y_1}}\rho_{x_k y_1}$. The obtained correlation $\rho'_{x_k y_1}$ should have the same value in both equations, hence, we deduce the equality $\frac{\rho_{x_i y_1}}{\rho'_{x_i y_1}} = \frac{\rho_{x_j y_1}}{\rho'_{x_j y_1}}$. On the other hand, from $\rho_{x_i x_j} = \rho_{x_i y_1}\rho_{x_j y_1}$, we have $\frac{\rho_{x_i y_1}}{\rho'_{x_i y_1}} = \frac{\rho'_{x_j y_1}}{\rho_{x_j y_1}}$. By these two equations we may conclude $\rho_{x_i y_1} = \rho'_{x_i y_1}$, a contradiction. Hence, in this case, given $\Sigma_{\mathbf{x}}$ we cannot further vary the edge-weights. Second, consider the case, where x_i is connected to y_1 through several observables. Then, instead of x_i, we can simply consider the observable that is adjacent to y_1, say, x'_i and follows the previous steps to obtain the result. Hence, in general if three nodes are connected to each other through separate paths and have a common ancestor y_1, then the pairwise correlations between the hidden nodes and each of the observables remain fixed.

(3) Consider two adjacent latent nodes y_1 and y_2. By minimality assumption and having a tree structure, it can be seen that there are at least two observable for each of the latent nodes that share a common latent parent. Let's assign x_i and x_j to a common ancestor y_1 while x_k

and x_k are descendant to y_2. Considering x_i, x_j, and x_k, who share a common parent y_1 (x_k is connected to y_1 through y_2), using arguments on case (2), we conclude that $\rho_{x_i y_1}$ and $\rho_{x_j y_1}$ should be fixed. Similarly, we can consider x_i, x_k, and x_l to show that $\rho_{x_k y_1}$ and $\rho_{x_l y_1}$ are fixed. Now, by considering any observable pair that go through both y_1 and y_2 the result follows. For example, considering $\rho_{x_i x_k} = \rho_{x_i y_1}\rho_{y_1 y_2}\rho_{x_k y_1}$, we can see that since given $\rho_{x_i x_k}$, both $\rho_{x_i y_1}$ and $\rho_{x_k y_1}$ are determined, so $\rho_{y_1 y_2}$ should be determined as well. This completes the first part of the proof.

Second, note that one may easily show that $I(\mathbf{X}, \tilde{\mathbf{Y}}) = 1/2 \log \frac{|\Sigma_{\mathbf{x}}||\Sigma_{\tilde{\mathbf{y}}}|}{|\Sigma_{\mathbf{x}\tilde{\mathbf{y}}}|}$. Now, since $p_{\mathbf{X},\tilde{\mathbf{Y}}}$ induces a latent Gaussian tree and $p_{\tilde{\mathbf{Y}}}$ is its marginalized density after summing out the random vector $\mathbf{X}$. By [31], we know that $|\Sigma_{\mathbf{X},\tilde{\mathbf{Y}}}| = \prod_{(i,j)\in E}(1-\rho_{i,j}^2)$, where $\rho_{i,j}$ are the pairwise correlations, between two adjacent variables (hidden or observable) in a latent Gaussian tree. Now, since the observables are only leaves, by summing them out we end with another Gaussian tree consisting of only latent variables. Thus, again by [31] we know $|\Sigma_{\tilde{\mathbf{Y}}}| = \prod_{(i,j)\in E_y}(1-\rho_{i,j}^2)$, where E' is the set of edges in the new Gaussian tree. Observe that all the common terms of the form $(1-\rho_{y_i y_j}^2)$, for some $(y_i, y_j) \in E$ will be canceled out with the terms in $|\Sigma_{\tilde{\mathbf{Y}}}|$. Hence, the mutual information has the following form $I(\mathbf{X}, \tilde{\mathbf{Y}}) = 1/2 \log \frac{|\Sigma|_{\mathbf{X}}}{\prod_{(x_i,y_j)\in E}(1-\rho_{x_i y_j}^2)}$. Now, to find each correlation value $\rho_{x_i y_j}$, for some X_i and Y_j, first consider the star model, with one hidden node, and three leaves, e.g., Figure 3.1. We can write: $\rho_{x_1 y}^2 = \frac{\rho_{x_1 x_2}\rho_{x_1 x_3}}{\rho_{x_2 x_3}}$, $\rho_{x_2 y}^2 = \frac{\rho_{x_1 x_2}\rho_{x_2 x_3}}{\rho_{x_1 x_3}}$, and $\rho_{x_3 y}^2 = \frac{\rho_{x_1 x_3}\rho_{x_2 x_3}}{\rho_{x_1 x_2}}$. For a general structure, if we replace $1 \leftarrow i$, $2 \leftarrow j_i$, and $3 \leftarrow k_i$, we conclude that $\rho_{x_i y_j}^2 = \frac{\rho_{x_i x_{j_i}}\rho_{x_i x_{k_i}}}{\rho_{x_{j_i} x_{k_i}}}$, for any three distinct $i,\ j_i$ and k_i. As it may seem, there are many equations for computing $\rho_{x_i y_j}^2$, which all of these expressions should be equal, i.e., the covariance matrix Σ_x should be representable by a given latent tree model.

3.6.3 Proof of Theorem 3

Suppose the latent Gaussian tree has k latent variables,i.e., $\mathbf{Y} = [Y_1, Y_2, ..., Y_k]$. By adding back the sign variables the joint density $p_{\mathbf{XY}}$ becomes a Gaussian mixture model. One may model such

mixture as the summation of densities that are conditionally Gaussian, given sign vector.

$$p_{\mathbf{XY}}(\mathbf{x},\mathbf{y}) = \sum_{i=0}^{2^k-1} \eta_{\mathbf{B}_i} f_i(\mathbf{x},\mathbf{y}) \tag{3.15}$$

where each $\eta_{\mathbf{B}_i}$ captures the overall probability of the binary vector $\mathbf{B}_i = [b_{1i}, b_{2i}, ..., b_{ki}]$, with $b_{ji} \in \{0,1\}$. Here, $b_{ji} = 0$ is equivalent to having $b_{ji} = -1$. The terms $f_i(\mathbf{x},\mathbf{y})$ are conditional densities of the form $p(\mathbf{x},\mathbf{y}|\mathbf{B}_i)$

In order to characterize $I(\mathbf{X},\mathbf{Y})$, we need to find $p_{\mathbf{Y}}(\mathbf{y})$ in terms of $\eta_{\mathbf{B}_i}$ and conditional Gaussian densities as well. First, let's show that for any two hidden nodes y_i and y_j in a latent Gaussian tree, we have $E[y_i y_j] = \rho_{y_i y_j} b_i b_j$. The proof goes by induction: We may consider the structure shown in Figure 3.2(a) as a base, where we proved that $B_{12} = B_1^{(1)} B_2^{(1)}$. Then, assuming such result holds for any Gaussian tree with $k-1$ hidden nodes, we prove it also holds for any Gaussian tree with k hidden nodes. Let's name the newly added hidden node as y_k that is connected to several hidden and/or observable such that the total structure forms a tree. Now, for each newly added edge we assign $b_k b_{n_k}$, where $n_k \in N_k$ is one of the neighbors of y_k. Note that this assignment maintains the pairwise sign values between all previous nodes, since to find their pairwise correlations we go through y_k at most once, where upon entering/exiting y_k we multiply the correlation value by b_k, hence producing $b_k.b_k = 1$, so overall the pairwise correlation sign does not change. Note that the other pairwise correlation signs that do not pass through C_k remain unaltered. One may easily check that by assigning $b_k b_{n_k}$ to the sign value of each newly added edge we make y_k to follow the general rule, as well. Hence, overall we showed that $E[y_i y_j] = \rho_{y_i y_j} b_i b_j$ for any $y_i, y_j \in \mathbf{Y}$. This way we may write $\Sigma_{\mathbf{y}} = B\Sigma'_{\mathbf{y}}B$, where $\rho_{y_i y_j} \in \Sigma'_{\mathbf{y}}$ and $b_i \in B$ is $k \times k$ diagonal matrix. One may easily see that both B and its negation matrix $-B$ induce the same covariance matrix $\Sigma_{\mathbf{y}}$. As a result, if we define $\eta_{\bar{\mathbf{B}}_i}$ as a compliment of $\eta_{\mathbf{B}_i}$, we can write the mixture density $p_{\mathbf{Y}}(\mathbf{y})$ as follows,

$$p_{\mathbf{Y}}(\mathbf{y}) = \sum_{i=0}^{2^{k-1}-1} (\eta_{\mathbf{B}_i} + \eta_{\bar{\mathbf{B}}_i}) g_i(\mathbf{y}) \tag{3.16}$$

where the conditional densities can be characterized as $g_i(\mathbf{y}) = p(\mathbf{y}|\mathbf{B}_i) = p(\mathbf{y}|\bar{\mathbf{B}}_i)$. We know that $g_i(\mathbf{y}) = \int f_j(\mathbf{x}, \mathbf{y}) d\mathbf{x}$, where j may correspond to either $\mathbf{B}_i$ or $\bar{\mathbf{B}}_i$.

First, we need to show that the mutual information $I(\mathbf{X}, \mathbf{Y})$ is a convex function of $\eta_{\mathbf{B}_i}$ for all $i \in [0, 2^k - 1]$. By equality $I(\mathbf{X}, \mathbf{Y}) = h(\mathbf{X}) - h(\mathbf{X}|\mathbf{Y})$, and knowing that given $\Sigma_{\mathbf{x}}$ the entropy $h(\mathbf{X}) = 1/2 \log(2\pi e)^n |\Sigma_{\mathbf{x}}|$ is fixed, we only need to show that the conditional entropy $h(\mathbf{X}|\mathbf{Y})$ is a concave function of $\eta_{\mathbf{B}_i}$. Using definition of entropy and by replacing for $p_{\mathbf{XY}}$ and $p_{\mathbf{Y}}$ using equations (3.15) and (3.16), respectively, we may characterize the conditional entropy. By taking second order derivative, we deduce the following,

$$\begin{aligned} \frac{\partial^2 h(\mathbf{X}|\mathbf{Y})}{\partial^2 \eta_i \eta_j} = &- \int \int \frac{f_i(\mathbf{x}, \mathbf{y}) f_j(\mathbf{x}, \mathbf{y})}{p_{\mathbf{XY}}} d\mathbf{x} d\mathbf{y} \\ &+ \int \frac{\tilde{g}_i(\mathbf{y}) \tilde{g}_j(\mathbf{y})}{p_{\mathbf{Y}}} d\mathbf{y} \end{aligned} \tag{3.17}$$

where for simplicity of notations we write η_i instead of $\eta_{\mathbf{B}_i}$. Also, $\tilde{g}_i(\mathbf{y}) = \tilde{g}_{\bar{i}}(\mathbf{y}) = g_i(\mathbf{y})$ for $i \in [0, 2^{k-1} - 1]$. Note the following relation,

$$\begin{aligned} \int \int \frac{\tilde{g}_i(\mathbf{y}) f_j(\mathbf{x}, \mathbf{y}) p_{\mathbf{X}|\mathbf{Y}}}{p_{\mathbf{XY}}} d\mathbf{x} d\mathbf{y} &= \int \int \frac{\tilde{g}_i(\mathbf{y}) f_j(\mathbf{x}, \mathbf{y})}{p_{\mathbf{Y}}} d\mathbf{x} d\mathbf{y} \\ &= \int \frac{\tilde{g}_i(\mathbf{y})}{p_{\mathbf{Y}}} (f_j(\mathbf{x}, \mathbf{y}) d\mathbf{x}) d\mathbf{y} \\ &= \int \frac{\tilde{g}_i(\mathbf{y}) \tilde{g}_j(\mathbf{y})}{p_{\mathbf{Y}}} d\mathbf{y} \end{aligned} \tag{3.18}$$

The same procedure can be used to show,

$$\int \int \frac{\tilde{g}_j(\mathbf{y}) f_i(\mathbf{x}, \mathbf{y}) p_{\mathbf{X}|\mathbf{Y}}}{p_{\mathbf{XY}}} d\mathbf{x} d\mathbf{y} = \int \frac{\tilde{g}_i(\mathbf{y}) \tilde{g}_j(\mathbf{y})}{p_{\mathbf{Y}}} d\mathbf{y} \tag{3.19}$$

By equalities shown in (3.18) and (3.19), it is straightforward that (3.17) can be turn into the following,

$$\begin{aligned} h_{ij} = \frac{\partial^2 h(\mathbf{X}|\mathbf{Y})}{\partial^2 \eta_i \eta_j} = &- \int \int \frac{1}{p_{\mathbf{XY}}} [f_i(\mathbf{x}, \mathbf{y}) - \tilde{g}_i(\mathbf{y}) p_{\mathbf{X}|\mathbf{Y}}] \times \\ &[f_j(\mathbf{x}, \mathbf{y}) - \tilde{g}_j(\mathbf{y}) p_{\mathbf{X}|\mathbf{Y}}] d\mathbf{x} d\mathbf{y} \end{aligned} \tag{3.20}$$

The matrix $H = [h_{ij}],\ i, j \in [0, 2^k - 1]$ characterizes the Hessian matrix the conditional entropy $h(\mathbf{X}|\mathbf{Y})$. To prove the concavity, we need to show H is non-positive definite. Define a non-zero real row vector $\mathbf{c} \in R^{2^k}$, then we need to form $\mathbf{c}H\mathbf{c}^T$ as follows and show that it is non-positive.

$$\begin{aligned}\mathbf{c}H\mathbf{c}^T = &- \int\int \frac{1}{p_{\mathbf{XY}}} \sum_{i=0}^{2^k-1} \sum_{j=0}^{2^k-1} c_i c_j [f_i(\mathbf{x}, \mathbf{y}) - \tilde{g}_i(\mathbf{y}) p_{\mathbf{X}|\mathbf{Y}}] \\ &\qquad [f_j(\mathbf{x}, \mathbf{y}) - \tilde{g}_j(\mathbf{y}) p_{\mathbf{X}|\mathbf{Y}}] d\mathbf{x} d\mathbf{y} \\ = &- \int\int \frac{1}{p_{\mathbf{XY}}} [\sum_{i=0}^{2^k-1} c_i (f_i(\mathbf{x}, \mathbf{y}) - \tilde{g}_i(\mathbf{y}) p_{\mathbf{X}|\mathbf{Y}})]^2 d\mathbf{x} d\mathbf{y} \\ &\leq 0 \end{aligned} \tag{3.21}$$

Now that we showed the concavity of the conditional entropy with respect to η_i, we only need to find the optimal solution. The formulation is defined in (3.22), where λ is the Lagrange multiplier.

$$L = h(\mathbf{X}|\mathbf{Y}) - \lambda \sum_{i=0}^{2^k-1} \eta_i \tag{3.22}$$

by taking derivative with respect to η_i, we may deduce the following,

$$\begin{aligned}\frac{\partial L}{\partial \eta_i} = &- \int\int f_i(\mathbf{x}, \mathbf{y}) \log p_{\mathbf{XY}} d\mathbf{x} d\mathbf{y} \\ &+ \int \tilde{g}_i(\mathbf{y}) \log p_{\mathbf{Y}} d\mathbf{y} - \lambda \\ = &- \int\int f_i(\mathbf{x}, \mathbf{y}) \log p_{\mathbf{X}|\mathbf{Y}} d\mathbf{x} d\mathbf{y} - \lambda \end{aligned} \tag{3.23}$$

where the last equality is due to $\tilde{g}_i(\mathbf{y}) = \int f_i(\mathbf{x}, \mathbf{y}) d\mathbf{x}$. One may find the optimal solution by solving $\partial L / \partial \eta_i = 0$ for all $i \in [0, 2^k - 1]$, which results in showing that $-\int\int [f_i(\mathbf{x}, \mathbf{y}) - f_j(\mathbf{x}, \mathbf{y})] \log p_{\mathbf{X}|\mathbf{Y}} d\mathbf{x} d\mathbf{y} = 0$, for all $i, j \in [0, 2^k - 1]$. In order to find the joint Gaussian density $f_i(\mathbf{x}, \mathbf{y})$, observe that we should compute the exponent $[\mathbf{xy}] \Sigma_{\mathbf{xy}}^{-1} [\mathbf{xy}]'$. Since, we are dealing with a latent Gaussian tree, the structure of $U = \Sigma_{\mathbf{xy}}^{-1}$ can be summarized into four blocks as follows [55]. $U_{\mathbf{x}}$ that has diagonal and off-diagonal entries u_{x_i} and $u_{x_i x_j}$, respectively, and not depending on the edge-signs; $U_{\mathbf{xy}}$, with nonzero elements $u_{x_i y_j}$ showing the edges between x_i and particular y_j and depending on correlation signs; $[U_{\mathbf{xy}}]^T$; $U_{\mathbf{y}}$, with nonzero off diagonal elements $u_{y_i y_j}$ that

are a function of edge-sign values, while the diagonal elements u_{y_i} are independent of edge-sign values. One may show,

$$\begin{aligned}[\mathbf{xy}]\Sigma_{\mathbf{xy}}^{-1}[\mathbf{xy}]' &= [\sum_{i=1}^{n} x_i^2 u_{x_i} + \sum_{i=1}^{k} y_i^2 u_{y_i}] \\ &+ 2[\sum_{n_{y_1}^x} x_i y_1 u_{x_i y_1} + ... + \sum_{n_{y_k}^x} x_i y_k u_{x_i y_k}] \\ &+ 2\sum_{(i,j)\in E_X} x_i x_j u_{x_i x_j} + 2\sum_{(i,j)\in E_Y} y_i y_j u_{y_i y_j} \\ &= t + 2\sum_{i=1}^{k} p_i + 2s + 2\sum_{(i,j)\in E_Y} y_i y_j u_{y_i y_j} \end{aligned} \tag{3.24}$$

where $n_{y_i}^x$ are the observed neighbors of y_i, and E_Y is the edge set corresponding only to hidden nodes, i.e., those hidden nodes that are adjacent to each other. E_X can be defined similarly, with $s = \sum_{(i,j)\in E_X} x_i x_j u_{x_i x_j}$. Also $p_j = \sum_{n_{y_j}^x} x_i y_j u_{x_i y_j}$. Suppose $f_i(\mathbf{x}, \mathbf{y})$ and $f_j(\mathbf{x}, \mathbf{y})$ are different at l sign values $\{i_1, ..., i_l\} \in L$. Let's write,

$$\begin{aligned}\sum_{(i,j)\in E_Y} y_i y_j u_{y_i y_j} &= \sum_{\substack{(i,j)\in E_Y \\ i,j\in L \text{ or } i,j\notin L}} y_i y_j u_{y_i y_j} \\ &+ \sum_{\substack{(i,j)\in E_Y \\ i \text{ or } j\in L}} y_i y_j u_{y_i y_j} \\ &= q + q' \end{aligned} \tag{3.25}$$

Hence, we divide the summation $\sum_{(i,j)\in E_Y} y_i y_j u_{y_i y_j}$ into two parts q and q'. Suppose $\eta_i = 1/2^k$ for all $i \in [0, 2^k - 1]$. We may form $f_i(\mathbf{x}, \mathbf{y}) - f_j(\mathbf{x}, \mathbf{y})$ as follows,

$$\begin{aligned} f_i(\mathbf{x}, \mathbf{y}) - f_j(\mathbf{x}, \mathbf{y}) \propto & e^{-t/2+s+q+\sum_{i\notin L} p_i} \\ & \times \left[e^{q'+\sum_{i\in L} p_i} - e^{-q'-\sum_{i\in L} p_i}\right] \end{aligned}$$

By negating all $y_{i_1}, ..., y_{i_l}$ into $-y_{i_1}, ..., -y_{i_l}$, it is apparent that t, $\sum_{i\notin L} p_i$, and s do not change. Also, the terms in q either remain intact or doubly negated, hence, overall q remains intact also. However, by definition, $p_i, i \in L$ will be negated, hence overall the sum $\sum_{i\in L} p_i$ will be negated.

The same thing holds true for q', since exactly one variable y_i or y_j in the summation, will change its sign, so q' also will be negated. Overall, we can see that by negating $y_{i_1}, ..., y_{i_l}$, we will negate $f_i - f_j$. It remains to show that such negation does not impact $p_{\mathbf{X}|\mathbf{Y}}$. Note that since $p_{\mathbf{XY}}$ includes all 2^k sign combinations and all of $f_i(\mathbf{x}, \mathbf{y})$ are equi-probable since we assumed $\eta_i = 1/2^k$ so $p_{\mathbf{XY}}$ is symmetric with respect to η_i, and such transformation on $y_{i_1}, ..., y_{i_l}$ does not impact the value of $p_{\mathbf{XY}}$, since by such negation we simply switch the position of certain Gaussian terms $f_i(\mathbf{x}, \mathbf{y})$ with each other.

For $p_{\mathbf{y}}$, we should first compute the term $\mathbf{y}\Sigma_{\mathbf{y}}^{-1}\mathbf{y}'$. We know $\Sigma_{\mathbf{y}} = B\Sigma'_{\mathbf{y}}B$, so $\Sigma_Y^{-1} = B^{-1}\Sigma_{\mathbf{y}}'^{-1}B^{-1} = B\Sigma_{\mathbf{y}}'^{-1}B$ (note, $\Sigma_{\mathbf{y}}$ does not necessarily induce a tree structure). We have,

$$\mathbf{y}\Sigma_{\mathbf{y}}^{-1}\mathbf{y}' = \sum_{i=1}^{k} w_{ii}y_i^2 + 2\sum_{i,j\&i<j} w_{ij}y_iy_jb_ib_j$$

From this equation, we may interpret the negation of $y_{i_1}, ..., y_{i_l}$, simply as negation of $b_{i_1}, ..., b_{i_l}$. Hence, since $p_{\mathbf{y}}$ includes all sign combinations, hence, such transformation only permute the terms $\tilde{g}_i(\mathbf{y})$, so $p_{\mathbf{y}}$ remains fixed. Hence, overall $p_{\mathbf{X}|\mathbf{Y}}$ remains unaltered. As a result, we show that for any given point in the integral $\int\int(f_i(\mathbf{x}, \mathbf{y}) - f_j(\mathbf{x}, \mathbf{y}))\log p_{\mathbf{X}|\mathbf{Y}}d\mathbf{x}d\mathbf{y}$ we can find its negation, hence making the integrand an odd function, and the corresponding integral zero. Hence, making the solution $\eta_i = 1/2^k$, for all $i \in [0, 2^k - 1]$ an optimal solution.

The only thing remaining is to show that from $\eta_i = 1/2^k$ we may conclude that $\pi_j = 1/2$ for all $j \in [1, k]$. By definition, we may write,

$$\eta_i = \prod_{j=1}^{k} \pi_j^{b_{ji}}(1 - \pi_j)^{1-b_{ji}}$$

where $b_{ji} \in B_i$. Assume all $\eta_i = 1/2^k$. Consider η_1 and find η_{i^*} such that the two are different in only one expression, say at the l-th place. Since, all η_i are equal, one may deduce $1 - \pi_l = \pi_l$ so $\pi_l = 1/2$. Note that such η_{i^*} can always be found since η_i's are covering all possible combinations of k-bit vector. Now, find another η_{j^*}, which is different from η_1 at some other spot, say l', again using similar arguments, we may show $\pi_{l'} = 1/2$. This can be done k times to show that, if all $\eta_i = 1/2^k$, then $\pi_1 = ... = \pi_k = 1/2$. This completes the proof.

3.6.4 Proof of Theorem 4

The signal model can be directly written as follows,

$$\mathbf{Y}^{(l)} = A_{\mathbf{B}^{(l,l+1)}}\mathbf{Y}^{(l+1)} + \mathbf{Z}^{(l)} \tag{3.26}$$

Here, we show the codebook generation scheme to generate $\mathbf{Y}^{(l)}$ from $\mathbf{Y}^{(l+1)}$. Note that $\mathbf{Y}^{(l)}$ is a vector consisting of the variables $Y_i^{(l)}$. Also, $\mathbf{Y}^{(l+1)}$ is a vector consisting of variables $Y_i^{(l+1)}$. The proof relies on the procedure taken in [16]. Note that our scheme should satisfy the following constraints,

$1)(Y_i^{(l)})^N \perp (Y_j^{(l)})^N|\tilde{\mathbf{Y}}^{(l+1)} \ \ (i \neq j)$

$2)(\mathbf{Y}^{(l)})^N \perp \mathbf{B}^{(l+1)}$

$3)P_{(\mathbf{Y}^{(l)})^N} = \prod_{t=1}^{N} P_{\mathbf{Y}^{(l)}}(\mathbf{y}_t^{(l)})$

$4)|\mathbf{Y}^{(l+1)}| = 2^{NR_{\mathbf{Y}^{(l+1)}}}$

$5)|\mathbf{B}^{(l+1)}| = 2^{NR_{\mathbf{B}^{(l+1)}}}$

$6)||q_{(\mathbf{Y}^{(l)})^N} - \prod_{t=1}^{N} P_{\mathbf{Y}^{(l)}}(\mathbf{y}_t^{(l)})||_{TV} < \epsilon$

where the first constraint is due to the conditional independence assumption characterized in the signal model (3.26). The second one is to capture the intrinsic ambiguity of the latent Gaussian tree to capture the sign information. Condition $3)$ is due to independence of joint densities $P_{\mathbf{Y}^l}(\mathbf{Y}_t^l)$ at each time slot t. Conditions $4)$ and $5)$ are due to corresponding rates for each of the inputs $\mathbf{Y}^{(l+1)}$ and $\mathbf{B}^{(l+1)}$. And finally, condition $6)$ is the synthesis requirement to be satisfied. First, we generate a codebook $\mathcal{C}$ of $\tilde{y}^N$ sequences, with indices $y \in C_Y = \{1, 2, ..., 2^{NR_{\mathbf{Y}^{(l+1)}}}\}$ and $b \in C_B = \{1, 2, ..., 2^{NR_{\mathbf{B}^{(l+1)}}}\}$ according to the explained procedure in Algorithm 2. The codebook $\mathcal{C}$ consists of all combinations of the sign and latent variables codewords, i.e., $|\mathcal{C}| = |C_Y| \times |C_B|$. We construct the joint density $\gamma_{(\mathbf{Y}^{(l)})^N, \mathbf{Y}^{(l+1)}, \mathbf{B}^{(l+1)}}$ as depicted by Figure 3.14,

The indices y and b are chosen independently and uniformly from the codebook $\mathcal{C}$. As can be seen from Figure 3.14, for each $\mathbf{B}_t^{(l)} = \mathbf{b}_t^{(l)}$ the channel $P_{Y^l|\tilde{Y}}$ is in fact consists of n independent

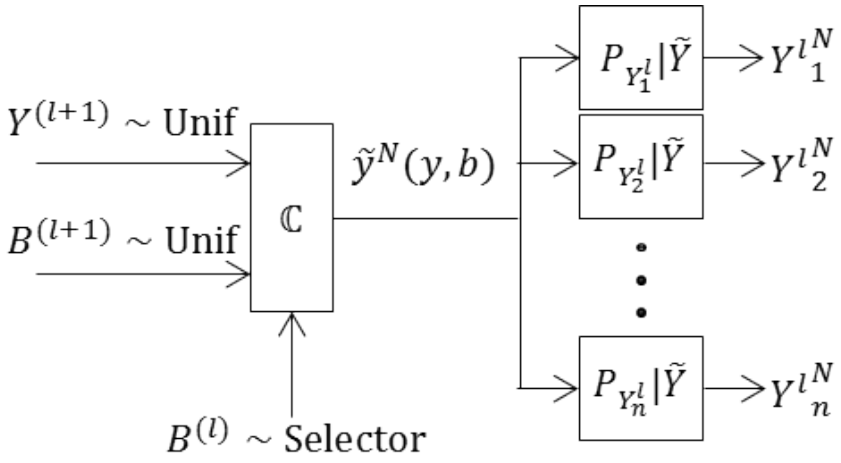


FIGURE 3.14: Construction of the joint density $\gamma_{(\mathbf{Y}^{(l)})^N,\mathbf{Y}^{(l+1)},\mathbf{B}^{(l+1)}}$

channels $P_{Y_i^l|\tilde{Y}},\ i \in \{1,2,...,n\}$. The joint density is as follows,

$$\gamma_{(\mathbf{Y}^{(l)})^N,\mathbf{Y}^{(l+1)},\mathbf{B}^{(l+1)}} = \frac{1}{|C_Y||C_B|}[\prod_{t=1}^{N} P_{\mathbf{Y}^l}(\mathbf{Y}_t^l|\tilde{y}_t(y,b))]$$

Note that $\gamma_{(\mathbf{Y}^{(l)})^N,\mathbf{Y}^{(l+1)},\mathbf{B}^{(l+1)}}$ already satisfies the constraints $1)$, $4)$, and $5)$ by construction. Next, we need to show that it satisfies the constraint $6)$. The marginal density $\gamma_{(\mathbf{Y}^{(l)})^N}$ can be deduced by the following,

$$\gamma_{(\mathbf{Y}^{(l)})^N} = \frac{1}{|C_Y||C_B|}\sum_{y\in C_Y}\sum_{b\in C_B}[\prod_{t=1}^{N} P_{\mathbf{Y}^{(l)}}(\mathbf{Y}_t^{(l)}|\tilde{y}_t(y,b))]$$

We know if $R_{\mathbf{B}^{(l+1)}} + R_{\mathbf{Y}^{(l+1)}} \geq I[\mathbf{Y}^{(l+1)},\mathbf{B}^{(l+1)};\mathbf{Y}^{(l)}|\mathbf{B}^{(l)}]$, then by soft covering lemma [16] we have,

$$\lim_{n\to\infty} E||\gamma_{(\mathbf{Y}^{(l)})^N} - \prod P_{\mathbf{Y}^{(l)}}||_{TV} = 0 \tag{3.27}$$

which shows that $\gamma_{(\mathbf{Y}^{(l)})^N}$ satisfies constraint $6)$. For simplicity of notations we use $\prod P_{\mathbf{Y}^{(l)}}$ instead of $\prod_{t=1}^{N} P_{\mathbf{Y}^{(l)}}(\mathbf{Y}_t^{(l)})$, since it can be understood from the context. Next, let's show that $\gamma_{(\mathbf{Y}^{(l)})^N}$, *nearly* satisfies constraints $2)$ and satisfies $3)$. We need to show that as N grows the synthesized density $\gamma_{(\mathbf{Y}^{(l)})^N,\mathbf{B}^{(l+1)}}$ approaches $\frac{1}{|C_B|}\prod P_{\mathbf{Y}^{(l)}}$, in which the latter satisfies both $2)$ and $3)$. In particular, we need to show that the total variation $E||\gamma_{(\mathbf{Y}^{(l)})^N,\mathbf{B}^{l+1}} - \frac{1}{|C_B|}\prod P_{\mathbf{Y}^{(l)}}||$ vanishes as

N grows. After taking several algebraic steps similar to the ones in [16], we should equivalently show that the following term vanishes, as $N \to \infty$,

$$\frac{1}{|C_B|}\sum_{b\in C_B} E||\gamma_{(\mathbf{Y}^{(l)})^N,\mathbf{B}^{l+1}=\mathbf{b}} - \prod P_{\mathbf{Y}^{(l)}}||_{TV} \tag{3.28}$$

Note that given any fixed $b \in C_B$ the number of Gaussian codewords is $|C_Y| = 2^{NR_{\mathbf{Y}^{(l+1)}}}$. Also, one can check by the signal model defined in (3.26) that the statistical properties of the output vector $\mathbf{Y}^{(l)}$ given any fixed sign value $b \in C_B$ does not change. Hence, for sufficiently large rates, i.e., $R_{\mathbf{Y}^{(l+1)}} \geq I[\mathbf{Y}^{(l+1)};\mathbf{Y}^{(l)}|\mathbf{B}^{(l)}]$, and by soft covering lemma, the term in the summation in (3.28) vanishes as N grows. So overall the term shown in (3.28) vanishes. This shows that in fact $\gamma_{(\mathbf{Y}^{(l)})^N}$ *nearly* satisfies the constraints $2)$ and $3)$. Hence, let's construct another distribution using $\gamma_{(\mathbf{Y}^{(l)})^N,\mathbf{Y}^{(l+1)},\mathbf{B}^{(l+1)}}$. Define,

$$q_{(\mathbf{Y}^{(l)})^N,\mathbf{Y}^{(l+1)},\mathbf{B}^{(l+1)}} = \frac{1}{|C_B|}(\prod P_{\mathbf{Y}^{(l)}})\gamma_{\mathbf{Y}^{(l+1)}|(\mathbf{Y}^{(l)})^N,\mathbf{B}^{(l+1)}}$$

It is not hard to see that such density satisfies $1)-5)$. We only need to show that it satisfies $6)$ as well. We have,

$$\begin{aligned}
&||q_{(\mathbf{Y}^{(l)})^N} - \prod P_{\mathbf{Y}^{(l)}}||_{TV} \\
&\leq ||q_{(\mathbf{Y}^{(l)})^N} - \gamma_{(\mathbf{Y}^{(l)})^N}||_{TV} + ||\gamma_{(\mathbf{Y}^{(l)})^N} - \prod P_{\mathbf{Y}^{(l)}}||_{TV} \\
&\leq ||q_{(\mathbf{Y}^{(l)})^N,\mathbf{Y}^{(l+1)},\mathbf{B}^{(l+1)}} - \gamma_{(\mathbf{Y}^{(l)})^N,\mathbf{Y}^{(l+1)},\mathbf{B}^{(l+1)}}||_{TV} + \epsilon_N \qquad (3.29)\\
&= ||q_{(\mathbf{Y}^{(l)})^N,\mathbf{B}^{(l+1)}} - \gamma_{(\mathbf{Y}^{(l)})^N,\mathbf{B}^{(l+1)}}||_{TV} + \epsilon_N \qquad (3.30)\\
&= ||\frac{1}{|C_B|}(\prod P_{\mathbf{Y}^{(l)}}) - \gamma_{(\mathbf{Y}^{(l)})^N,\mathbf{B}^{(l+1)}}||_{TV} + \epsilon_N \qquad (3.31)
\end{aligned}$$

where $\epsilon_N = ||\gamma_{(\mathbf{Y}^{(l)})^N} - \prod P_{\mathbf{Y}^{(l)}}||_{TV}$. Both terms in (3.31) vanish as N grows, due to (3.28) and (3.27), respectively. Note that, (3.29) is due to [16, Lemma V.I]. Also, (3.30) is due to [16, Lemma V.II], by considering the terms $q_{(\mathbf{Y}^{(l)})^N,\mathbf{Y}^{(l+1)},\mathbf{B}^{(l+1)}}$ and $\gamma_{(\mathbf{Y}^{(l)})^N,\mathbf{Y}^{(l+1)},\mathbf{B}^{(l+1)}}$ as the outputs of a unique channel specified by $\gamma_{\mathbf{Y}^{(l+1)}|(\mathbf{Y}^{(l)})^N,\mathbf{B}^{(l+1)}}$, with inputs $p_{(\mathbf{Y}^{(l)})^N,\mathbf{B}^{(l+1)}}$ and $\gamma_{(\mathbf{Y}^{(l)})^N,\mathbf{B}^{(l+1)}}$, respectively.

Finally, note that we synthesize each $\mathbf{Y}^{(l)}$ for a given $\mathbf{B}^{(l)} = \mathbf{b}$. Hence, to obtain the overall statistics we have $q_{\mathbf{Y}^{(l)}} = \sum_b q_{\mathbf{Y}^{(l)}|\mathbf{B}^{(l)}} p(\mathbf{B}^{(l)} = \mathbf{b})$, where the summation is over all possible sign combinations for layer $\mathbf{Y}^{(l)}$, which equals to 2^{k_l}. Certainly, this number becomes exponentially large if k_l is large. However, note that as $N \to \infty$ each synthesized output (for each given $\mathbf{B}^{(l)} = \mathbf{b}$) become arbitrarily close to zero. Hence, overall $q_{\mathbf{Y}^{(l)}}$ becomes arbitrarily close to the desired statistics. This is also the case for the overall latent Gaussian tree, i.e., for L capturing the total number of layers, at each layer we can generate an output with vanishing total variation distance from the desired statistics, hence overall the final output statistics becomes arbitrarily close to the desired output statistics.

This completes the achievability proof.

3.6.5 Proof of Lemma 7

First, we need to change the latent tree structure in a way similar to Figure 3.10. We start from the standard latent structure, and at each layer we seek for those latent nodes that are at the same layer and they are neighbors. For each pair of adjacent nodes, we move the one that is further away from the top layer to a new added layer below the current one. Hence, make a new layer of latent nodes. We iterate this step until we reach the bottom layer. This way, we face different groups of observables being synthesized at different layers.

Define $\mathbf{X}^{(l)}$, $\mathbf{Y}^{(l)}$ and $\mathbf{B}^{(l)}$ as the set of observables, latent nodes and sign variables at layer l, respectively. In this new setting layer $l = 0$ defines the observable layer, which only consists of remaining output variables, with no latent nodes. If the rates at each layer satisfy the inequalities in (3.13), then by Theorem 4 we know that as N increases, the simulated density $q_{(\mathbf{X}^{(l)})^N,(\mathbf{Y}^{(l)})^N}$ approaches to the desired density $\prod p_{(\mathbf{X}^{(l)}),(\mathbf{Y}^{(l)})}$. Suppose the first set of outputs are generated at layer L', then we know $\mathbf{X} = \bigcup_{l=0}^{L'} \mathbf{X}^{(l)}$. Each observable node $X_i^{(l)}$, for $l < L'$ has a latent ancestor at each layer $l < l' \leq L'$. We define $\mathbf{Y}'$ as the union of latent nodes containing all those latent ancestors. Basically, the vector $\mathbf{Y}'$ includes all the latent nodes $Y_j^{(l)}$ for $1 \leq l \leq L'$. We define $\mathbf{B}'$, similarly, i.e., those sign inputs related to the nodes in the set $\mathbf{Y}'$. With slightly

abuse of notation, define $\tilde{\mathbf{Y}} = \{\mathbf{Y}', \mathbf{B}'\}$, and $\tilde{\mathbf{Y}}^{(l)} = \{\mathbf{Y}^{(l)}, \mathbf{B}^{(l)}\}$, for all possible layers l. The scheme looks exactly as discussed previously, except that this time we need to keep track of corresponding generated outputs at each layer and match them together. In particular, consider the generated outputs $(\mathbf{X}^{(0)})^N$, which lie at the bottom layer. Each output is generated using a particular input vector $(\mathbf{Y}^{(1)})^N$, which in turn along with other possible outputs $(\mathbf{X}^{(1)})^N$ are generated by a unique input codeword $(\mathbf{Y}^{(2)})^N$ that lie at the second layer. This procedure moves from the bottom to the top layer, in order to match each generated output at the bottom layer with the correct output vectors at other layers. Note that the sign information will be automatically taken care of, since similar to the previous cases, at each layer $l+1$ and given each realization of the sign vector $\mathbf{B}^{(l)} = \mathbf{b}^{(l)}$, the input vector $\mathbf{Y}^{(l+1)}$ will become Gaussian. We only need to show that the synthesize density regarding to such formed joint vectors approaches to the desired output density, as N grows.

By the underlying structure of latent tree, one may factorize the joint density $q_{\mathbf{X}^N,\tilde{\mathbf{Y}}^N} = q_{(\mathbf{X}^{(L')})^N,(\tilde{\mathbf{Y}}^{(L')})^N} \prod_{l=0}^{L'-1} q_{(\mathbf{X}^{(l)})^N|(\tilde{\mathbf{Y}}^{(l+1)})^N}$. Note that the desired joint density $p_{\mathbf{X},\tilde{\mathbf{Y}}}$ also induces the same latent Gaussian tree, hence, we may write, $p_{\mathbf{X}^N,\tilde{\mathbf{Y}}^N} = p_{(\mathbf{X}^{(L')})^N,(\tilde{\mathbf{Y}}^{(L')})^N} \prod_{l=0}^{L'-1} p_{(\mathbf{X}^{(l)})^N|(\tilde{\mathbf{Y}}^{(l+1)})^N}$. However, by our synthesis scheme shown in Figure 3.14, one may argue that $\prod_{l=0}^{L'-1} q_{(\mathbf{X}^{(l)})^N|(\tilde{\mathbf{Y}}^{(l+1)})^N} = \prod_{l=0}^{L'-1} p_{(\mathbf{X}^{(l)})^N|(\tilde{\mathbf{Y}}^{(l+1)})^N} = \prod_{l=0}^{L'-1} \prod p_{\mathbf{X}_t^{(l)}|\tilde{\mathbf{Y}}_t^{(l+1)}}$. By summing out $(\mathbf{B}^{(L')})^N$ from both densities $p_{\mathbf{X}^N,\tilde{\mathbf{Y}}^N}$ and $q_{\mathbf{X}^N,\tilde{\mathbf{Y}}^N}$, we may replace $p_{(\mathbf{X}^{(L')})^N,(\tilde{\mathbf{Y}}^{(L')})^N}$ with $p_{(\mathbf{X}^{(L')})^N,(\mathbf{Y}^{(L')})^N}$ and $q_{(\mathbf{X}^{(L')})^N,(\tilde{\mathbf{Y}}^{(L')})^N}$ with $q_{(\mathbf{X}^{(L')})^N,(\mathbf{Y}^{(L')})^N}$, since only these terms in the equations depend on the sign vector at layer L', i.e., $(\mathbf{B}^{(L')})^N$. Now, by previous arguments for the synthesized and desired density at layer L', we know that the total variation distance $||q_{(\mathbf{X}^{(L')})^N,(\mathbf{Y}^{(L')})^N} - \prod p_{\mathbf{X}_t^{(L')},\mathbf{Y}_t^{(L')}}||_{TV}$ goes to zero as N grows. Hence, one may simply deduce that $||q_{\mathbf{X}^N,\tilde{\mathbf{Y}}^N/(\mathbf{B}^{(L')})^N} - \prod p_{\mathbf{X}_t,\tilde{\mathbf{Y}}_t/\mathbf{B}_t^{(L')}}||_{TV} = ||(q_{(\mathbf{X}^{(L')})^N,(\mathbf{Y}^{(L')})^N} - \prod p_{\mathbf{X}_t^{(L')},\mathbf{Y}_t^{(L')}}) \prod_{l=0}^{L'-1} \prod p_{\mathbf{X}_t^{(l)}|\tilde{\mathbf{Y}}_t^{(l+1)}}||_{TV}$ goes to zero as N grows. Due to [16, Lemma V.I], we know $||q_{\mathbf{X}^N} - \prod p_{\mathbf{X}_t}||_{TV} \leq ||q_{\mathbf{X}^N,\tilde{\mathbf{Y}}^N/(\mathbf{B}^{(L')})^N} - \prod p_{\mathbf{X}_t,\tilde{\mathbf{Y}}_t/\mathbf{B}_t^{(L')}}||_{TV} < \epsilon$, and as N grows. This completes the proof.

Chapter 4

Algebraic Properties of Solutions to Common Information of Gaussian Graphical Models

We formulate Wyner's common information for random vectors $\mathbf{x} \in \mathbb{R}^n$ with joint Gaussian density. We show that finding the common information of Gaussian vectors is equivalent to maximizing a log-determinant of the additive Gaussian noise covariance matrix. We coin such optimization problem as a constrained minimum determinant factor analysis (CMDFA) problem. The convexity of such problem with necessary and sufficient conditions on CMDFA solution is shown. We study the algebraic properties of CMDFA solution space, through which we study two extreme Gaussian graphical models, namely, *latent Gaussian stars*, and *explicit Gaussian chains*. Interestingly, we show that depending on pairwise covariance values in a Gaussian graphical structure, one may not always end up with the same parameters and structure for CMDFA solution as those found via graphical learning algorithms.

This study has resulted in one research paper, [56].

4.1 Introduction

Wyner's Common information $C(X_1, X_2)$ characterizes the minimum amount of common randomness needed to approximate the joint density between a pair of random variables X_1 and X_2 to be $C(X_1, X_2) = \min_{\substack{P_Y \\ X_1 - Y - X_2}} I(X_1, X_2; Y)$, where $X_1 - Y - X_2$ represents conditional independence between X_1 and X_2, given Y, where the joint density function is sought to esnure such conditional independence, as well as the given joint density of X_1 and X_2. In other words, one may see Wyner's common information as the *optimal* way of generating random outputs, through which the number of common random bits to produce the desired output is minimized. Han and Verdu, in [15] along the same problem, define the notion of *resolvability* of a given channel, which is defined as the minimal required randomness to generate output statistics in terms of a vanishing total variation distance between the synthesized and prescribed joint densities. Resolvability of a

channel is found to be a very intuitive description of common randomness in our settings, since it can be related to channel quality in terms of its noise power, and the noisier the channel the less number of common random bits needed to simulate the output [15]. Also, Cuff in [16] completely characterized the achievable rate regions needed to synthesize a memoryless channel, where he also used the total variation distance metric to show the quality of the proposed scheme.

There are several works that extend the classical bi-variate synthesis problem in Wyner's study to more general scenarios. A lower bound on the generalized Wyner's common information is obtained in [49]. In [46–48], the authors aim to define the common information of n dependent random variables, to further address the same question in this setting. In fact, the authors characterize the closed form solution for common information of Gaussian vectors with homogeneous (i.e., equal) pairwise correlation values. They resort to the same scenario as Wyner [14] did, i.e., considering one random variable Y to define such common randomness. Also, in [51] the authors completely characterize the common information between two jointly Gaussian vectors, as a function of certain singular values that are related to both joint and marginal covariance matrices of two Gaussian random vectors. However, they still divide the random vector into two groups, which makes it similar to Wyner's scenario.

In many cases, introducing one latent variable is not enough to capture the entire common information. Here, we motivate the *multi-variate* common information problem. Consider the following motivating example, showing that we need at least two latent random variables to capture common information.

Example 8. Suppose we are given a zero mean Gaussian random vector $\{X_1, X_2, X_3, X_4\}$ forming a Markov chain $X_1 - X_2 - X_3 - X_4$ with corresponding correlation matrix (normalized covariance matrix) $\Sigma = [\rho_{ij}]$, where $\rho_{ij},\ (i,j) \in [1,4]$ is a pairwise correlation between X_i and X_j. Also, to ignore infeasible or trivial cases, we need to have $\rho_{ij} \in (-1,1)$ and non-zero. The correlation space of Gaussian trees is fully characterized in [43]. It is also shown that in order to have a chain the pairwise correlations ρ_{ij} are the product of correlation values for those variables along

the path from X_i to X_j [31]. Our objective is to show if it is possible to form a latent star-shape Gaussian graph by adding a single random variable Y given which all four X_i are conditionally independent. We only need to consider jointly Gaussian vector $(\mathbf{X}, Y)$, as it is shown in [48] that jointly Gaussian vector $(\mathbf{X}, Y)$ maximizes the conditional entropy $h(\mathbf{X}|Y)$ hence minimizing the mutual information $I(\mathbf{X}; Y)$. Assuming Y is a zero mean Gaussian random variable with variance σ_y^2, we may write the signal model $[X_1, X_2, X_3, X_4]' = [a_1, a_2, a_3, a_4]'Y + [Z_1, Z_2, Z_3, Z_4]'$, where the vector $A = [a_1, a_2, a_3, a_4]'$ is to be determined by given constraints in the problem and $\{Z_1, Z_2, Z_3, Z_4\}$ are independent zero mean Gaussian noises with variance $\sigma_{z_i}^2$. From such signal model we may easily see that $\Sigma = AA'\sigma_y^2 + \Sigma_z$, where $AA'\sigma_y^2$ is a rank one positive semi-definite matrix and Σ_z is diagonal, with diagonal elements $\sigma_{z_i}^2$. We may move Σ_z to the other part of equation to show the matrix $\Sigma - \Sigma_z$ is a rank one positive semi-definite matrix. Such Hermitian matrix has the following form,

$$\Sigma - \Sigma_z = \begin{pmatrix} t_1 & \rho_{12} & \rho_{12}\rho_{23} & \rho_{12}\rho_{23}\rho_{34} \\ \rho_{12} & t_2 & \rho_{23} & \rho_{23}\rho_{34} \\ \rho_{12}\rho_{23} & \rho_{23} & t_3 & \rho_{34} \\ \rho_{12}\rho_{23}\rho_{34} & \rho_{23}\rho_{34} & \rho_{34} & t_4 \end{pmatrix} \tag{4.1}$$

where due to the fact that $\Sigma - \Sigma_z$ is positive semi-definite and $\sigma_{z_i}^2$'s are non-negative, each $t_i = 1 - \sigma_{z_i}^2,\ i \in [1, 4]$ is between 0 and 1. Since the matrix in (4.1) is rank one, hence we may pick one of the rows as a basis for the row space of this matrix. One may see that by choosing either the first or second row as a basis, we end up setting $\rho_{23} = \pm 1$, which we know is an infeasible value. Due to symmetry of chain structure, similar answers will be deduced by setting the third or fourth row as a basis. Hence, overall we reached to a contradictory conclusion: the matrix $\Sigma - \Sigma_z$ cannot be a rank one matrix.

This simple case study shows that depending on the covariance matrix structure, we may need a Gaussian random vector (instead of a single random variable) to capture the common information among variables with certain structures.

Recently, Veld and Gastpar [50] characterized common information problem for this general setting, and formulated the problem as a specific instance of maximum determinant (*maxdet*) [57] problems. They also analytically computed the common information value for a specific set of Gaussian joint densities with *circulant* covariance matrices. Steeg *et. al*, in [45] defined *information sieve*, which is closely related to common information metric to represent deep latent Gaussian structures. They showed that in many applications, such as *Blind Source Separation* (BSS), the intrinsic latent structure consists of more than a single variable, and that a multi-variate notion of common information is necessary to discover all the latent Gaussian sources. In our previous works, [34, 35] we addressed such multi-variate latent structure in a special case of Gaussian trees. We proved that for such cases, the common information restricted to the underlying tree structure is equal to the mutual information between observed variables and the latent variables.

Similar to these works, we in this chapter first show that in a Gaussian case the common information problem is equivalent to minimizing the negative of log-determinant function of the additive Gaussian noise covariance matrix, under certain constraints. We show the relation of such problem to the classical *constrained minimum trace factor analysis* (CMTFA) problem [58–60], where the objective is to minimize the trace of an additive Gaussian noise covariance matrix. Therefore, we name the common information problem as *constrained minimum determinant factor analysis* (CMDFA). Rather than proposing a numerical algorithm for solving such convex programming problem (which as we discuss, there are certain algorithms for numerically solving maxdet problem), we focus on studying the algebraic features of the solution space of CMDFA problem in general, and specifically for several case studies, where Σ follows certain latent (or explicit) graphical structure.

This chapter is organized as follows. In section 4.2 we give a proper formulation of CMDFA problem. In section 4.3 we show the solution space of CMDFA problem, and study couple of special cases for $n = 3$. Finally, we conclude the chapter in section 4.4.

4.2 Problem Formulation

We may straightforwardly generalize Wyner's common information into multi-variable setting as follows,

$$C(\mathbf{X}) = \min_{p_{\mathbf{Y}}(\mathbf{y}), p(\mathbf{X}|\mathbf{Y})} I(\mathbf{X};\mathbf{Y}) \quad s.t. \;\; p(\mathbf{X}|\mathbf{Y}) = \prod_{i=1}^{n} p(X_i|\mathbf{Y}) \tag{4.2}$$

where $\mathbf{X} \in \mathbb{R}^n$ is a Gaussian vector with covariance matrix $\Sigma_{\mathbf{x}}$ (without loss of generality we can set $\mu_{\mathbf{x}}$ to a zero vector), and $\mathbf{Y} \in \mathbb{R}^k$ ($k \leq n$) is the auxiliary (latent) random vector (a single random variable, in a special case) capturing common randomness in $\mathbf{X}$. Also, $I(\mathbf{X};\mathbf{Y})$ captures the mutual information between these two vectors. The only constraint is the conditional independence of all $X_i \in \mathbf{X}$ given the latent vector $\mathbf{Y}$.

We know $I(\mathbf{X};\mathbf{Y}) = h(\mathbf{X}) - h(\mathbf{X}|\mathbf{Y})$, with $h(.)$ being the differential entropy, since given $\Sigma_{\mathbf{x}}$, the first term is fixed. The common information problem is equivalent to maximizing the conditional entropy $h(\mathbf{X}|\mathbf{Y})$ with conditional independence constraint. It is shown in [48] that a jointly Gaussian latent vector $\mathbf{Y}$ can maximize such quantity, hence, we can limit the search space of problem to Gaussian $p_{\mathbf{Y}}$'s. Let us define an affine model $\mathbf{X} = A\mathbf{Y} + \mathbf{Z}$, where A is $n \times k$ transition matrix and $Z \in \mathbb{R}^n$ is a zero mean additive Gaussian noise vector, with diagonal covariance matrix D (hence, all $z_i \in \mathbf{Z}$ are independent), where the diagonal elements d_i are the corresponding variances for each z_i. The noise elements are independent of the latent vector $\mathbf{Y}$. We assume that the generative (affine) model's parameters, i.e., the transition matrix A_G and the diagonal covariance matrix D_G are known to us, either a priori or through a specific learning algorithm [2, 44]. Using such affine mapping we also satisfy the conditional independence constraint. As a result, one may show that $I(\mathbf{X};\mathbf{Y}) = \frac{1}{2}\log\frac{|\Sigma_{\mathbf{x}}|}{|D|}$. We may re-write the common information

problem in (4.2) as follows,

$$\min_{D} -\log|D|,\ s.t.\quad \begin{cases} D \succ 0\ and\ diagonal \\ \Sigma_{\mathbf{x}} - D \succeq 0 \end{cases} \tag{4.3}$$

We coin the optimization problem in (4.3) as CMDFA. The matrix D has to be positive definite, i.e., all diagonal entries d_i should be positive. Otherwise, if for some i, d_i is zero, then we know $-\log|D| = -\log\prod_{i=1}^{n} d_i \rightarrow +\infty$, which maximizes the objective function. The second constraint is due to affine modeling: $\Sigma_{\mathbf{x}} = AA' + D$, where A' is the transpose of A. Therefore, $\Sigma_{\mathbf{x}} - D = AA' \succeq 0$. In particular, the rank of AA' is at most $k \leq n$. It can be easily shown that CMDFA is an instance of general class of optimization problems known as *max-det* problems [57]. Hence, several iterative algorithms proposed in [57] can be used to numerically find the solution for such optimization problem. In fact, a Matlab-based modeling system for convex optimization, known as CVX [61, 62] can be used to solve such problem. Rather, our goal is to study the algebraic properties of CMDFA solution space in general, and for certain Gaussian graphical structures.

From now on, for simplicity and without loss of generality, we assume $\Sigma_{\mathbf{x}}$ to be a correlation matrix, i.e., all $\sigma_{ij} \in \Sigma_{\mathbf{x}}$ are normalized to $\rho_{ij} = \sigma_{ij}/\sqrt{\sigma_{ii}\sigma_{jj}}$. As a result, due to the constraint $\Sigma_{\mathbf{x}} - D \succeq 0$, for all $d_i \in D$ we have $d_i < 1$. This would be fine, since it is shown [57][p. 3] that such problem is invariant under congruence transformations. Hence, once the solution D_1 for the correlation matrix is found, one may propose $D_2 = \Lambda^{1/2} D_1 \Lambda'^{1/2}$ (Λ is a diagonal matrix with $\lambda_i = \sigma_{ii}$) as a solution to the un-normalized CMDFA problem.

4.3 Main Results

In this section, we first give the necessary and sufficient conditions under which D^* can be the solution to CMDFA problem. The proof procedure is very similar to CMTFA proof proposed in [58, 59]. Then, we aim to characterize the solution in certain cases, where the Gaussian density $p_{\mathbf{X}}$ follows either a latent or explicit Gaussian tree structure.

In Theorem 5 whose proof can be found in Appendix 4.5.1 we characterize the conditions under which D^* is the solution to CMDFA.

Theorem 5. *The diagonal positive definite matrix D^* is a solution to CMDFA problem if and only if $|\Sigma - D^*| = 0$ and there exists a Gramian matrix $T = [t_{ij}] \succeq 0$, whose entries satisfy the following condition,*

$$\frac{1}{\mathbf{d}^*} = \sum_{i=1}^{n-k} \sum_{j=1}^{n-k} t_{ij} \mathbf{e}_i \mathbf{e}_j^* \tag{4.4}$$

where $1/\mathbf{d}^* = [1/d_1, ..., 1/d_n]'$ and $\mathbf{e}_i \in N(\Sigma - D^*)$ is a basis vector in null space of $\Sigma - D^*$. The notation $\mathbf{e}_i \mathbf{e}_j^* = [e_{i1} e_{j1}, ..., e_{in} e_{jn}]'$ is used for the Hadamard product of two basis vectors.

Remark 7. The rank deficiency constraint in Theorem 5, suggests the solution D^* to be always on the boundary $|\Sigma - D^*| = 0$. This is a necessary condition for CMDFA solution. Otherwise, assume D^* is such that $\Sigma - D^*$ is a full rank. As a result, all n principal minors of this matrix should be positive. However, we know that each of these principal minors are polynomial functions of d_i^*'s. We may propose another matrix $\tilde{D} = diag(d_1^* + \epsilon_1, ..., d_n^* + \epsilon_n)$, where $\epsilon \geq 0$ for all i. Due to smoothness of such polynomial functions, we can always find at least one ϵ_i (although very small) such that still all principal minors of $\Sigma - \tilde{D}$ remain positive, hence, keeping the matrix positive definite. However, now D^* cannot be CMDFA solution, since apparently $-\log|\tilde{D}| < -\log|D^*|$, a contradiction. Therefore, the rank of $\Sigma - D^*$ should be $k \leq n - 1$.

Remark 8. Then, one may question the existence of CMDFA solution, i.e., whether all postive definite matrices Σ can be decomposed into sum of $AA' + D$, where $AA' \succeq 0$ is rank deficient and $D^* \succ 0$ is diagonal. To show the existence of solution, define $\lambda_{min} > 0$ as the smallest eigenvalue of Σ. Then, considering the matrix $D = \lambda_{min} I \succ 0$, where I is $n \times n$ identity matrix, we know $\Sigma - D$ is both positive semi-definite and rank deficient (its minimum eigenvalue is zero). Hence, for any given matrix Σ the search space of CMDFA problem is nonempty.

In Theorem 6, whose proof can be found in Appendix 4.5.2 we take the same steps as in [59][Theorem 4] to show the uniqueness of CMDFA solution.

Theorem 6. *The CMDFA problem has a unique solution.*

In the remainder of the chapter we go through several cases to study their solution space regarding the CMDFA problem.

4.3.1 Star Structure

Suppose in the underlying affine model, the latent vector is a singleton Y, i.e., star structure. This can be modeled as

$$\begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} Y + \begin{bmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_n \end{bmatrix} \tag{4.5}$$

where $0 \leq a_i < 1,\ i \in [1, n]$. A special case for such model, with $n = 3$ is shown in Figure 4.1.

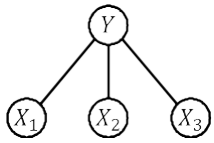


FIGURE 4.1: Star structure with $n = 3$ outputs

Using Theorem 5 we may be able to characterize the solution space of CMDFA problem as follows. Note that the pairwise correlations have the form $\rho_{ij} = a_i a_j,\ i \neq j \in [1, n]$. Basically, each a_i can be seen as an edge weight between a latent factor Y and the corresponding variable X_i.

For all $d_i \in D$ define $t_i = 1 - d_i$, where for CMDFA problem we know $0 \leq t_i < 1$.

In our previous studies [34–36], we showed that the value of each t_i is given and it is equal to a_i^2. Now, one can easily check that $rank(\Sigma - D) = 1$, also, $Trace(\Sigma - D) = \sum_{i=1}^{n} a_i^2 = \sum \lambda_i > 0$, where λ_i is an eigenvalue of $\Sigma - D$, but since $rank(\Sigma - D) = 1$, hence, $\lambda_2 = ... = \lambda_n = 0$, hence,

$\lambda_1 = \sum_{i=1}^{n} a_i^2 > 0$, so $\Sigma - D$ is positive-semidefinite in this case. Also, since $d_i = 1 - a_i^2 > 0$, hence $D \succ 0$. Therefore, such case lies on the feasible solution of CMDFA.

To check if this is indeed the solution of CMDFA, we first need to find the null space of $\Sigma - D$, which has rank $n-1$. By solving the system of equations $(\Sigma - D).\mathbf{V} = \mathbf{0}$, we deduce that the null space has the form $N(\Sigma - D) = \{\mathbf{V} | \sum_{i=1}^{n} a_i v_i = 0\}$. In other words, the vectors in null space have the form $\mathbf{V} = (v_1, v_2, ..., -\frac{1}{a_n} \sum_{i=1}^{n-1} a_i v_i)$.

One intuitive suggestion for the basis would be choosing the set of linearly independent vectors $e_1 = (1, 0, 0, ..., 0, -\frac{a_1}{a_n}), ..., e_n = (0, 0, ..., 0, 1, -\frac{a_{n-1}}{a_n})$. To find a Gramian matrix $T = [t_{ij}]$ satisfying(4.4), we obtain the following system of equalities

$$\begin{cases} t_{ii} = \dfrac{1}{1 - a_i^2},\ i \in [1, n-1] \\ 2 \sum_{i<j} t_{ij} a_i a_j = \dfrac{a_n^2}{1 - a_n^2} - \sum_{i=1}^{n-1} \dfrac{a_i^2}{1 - a_i^2},\ i, j \in [1, n-1] \end{cases} \tag{4.6}$$

Remark 9. Suppose, all $a_i = a$ are equal. This is the case considered in [46–48]. Then, using (4.6) we obtain $t_{ii} = \dfrac{1}{1 - a^2},\ i \in [1, n-1]$ and $\sum_{i<j} t_{ij} = -\dfrac{n-2}{2(1-a^2)},\ i, j \in [1, n-1]$. Simply putting all t_{ij}'s to be equal, gives $t_{ij} = -\dfrac{n-2}{n(n-1)(1-a^2)},\ i \neq j$. In this case $T = [t_{ij}]$ is a strictly digonally dominant matrix, since $|t_{kk}| > \sum_{i \neq k} |t_{ki}| = \dfrac{(n-2)^2}{n(n-1)(1-a^2)}$. And since all of its diagonal entries are positive, hence T is in fact positive semi-definite. As it can be seen this is a special case that satisfies the system of equalities and shows that CMDFA solution for a such affine model with single hidden variable (i.e., a star), is a star!

One may wonder if the above system of linear equations always holds regardless of given values for a_i's. In other words, does always exist a Gramian matrix T satisfying the following system of equalities? So that the CDMFA solution of a given Gaussian vector, which was generated using a star-generative and latent Gaussian graph, also ends up with a star? Through the following case study we show that this is not the case, even for the smallest star tree with $n = 3$ output variables.

4.3.2 CMDFA Solution Space for $n = 3$

Here, we consider a special case, where the set of output variables is a three dimensional vector $\mathbf{X} = \{X_1, X_2, X_3\}$. As we will, although this seems a small number of variables to begin with, but finding CMDFA solution proves to be a non-trivial task.

4.3.3 Star: Rank One Matrix

Suppose, $\Sigma - D$ is a rank one matrix, i.e., the latent vector is a singleton Y. Here, we draw an interesting conclusion, that the solution to CMDFA for such affine model, is not necessarily a star. This is shown in Theorem 7, whose proof can be found in Appendix 4.5.3.

Theorem 7. *For $n = 3$, and a rank one $\Sigma - D$ following the affine model in* (4.5)*, the CMDFA solution is also a star with the same parameters if and only if the following inequality holds,*

$$\begin{aligned} S_1 = \{s_1, s_2, s_3 | &(s_1 - s_2)^2 + (s_1 - s_3)^2 + (s_2 - s_3)^2 \\ &\leq s_1^2 + s_2^2 + s_3^2\} \end{aligned} \tag{4.7}$$

where $s_i = \dfrac{a_i^2}{1 - a_i^2},\ i \in [1, 3]$.

Hence, in some cases, despite the fact that a latent tree induces a star structure, but the CMDFA solution is not necessarily a star.

Remark 10. s_i's can be seen as Signal to Noise Ratio (SNR) of the three Gaussian channels in the affine model. And the feasible region can be re-written as $(\sqrt{s_1} - \sqrt{s_2})^2 \leq s_3 \leq (\sqrt{s_1} + \sqrt{s_2})^2$. The following figures show the feasible region in both different SNR and (Positive) Edge weights domains.

It is noteworthy to mention that the general case considered for example in [46] is a special case for this region, i.e., the diagonal line inside the region where $s_1 = s_2 = s_3$ or equivalently $|a_1| = |a_2| = |a_3|$.

Using previous remarks, since we know the CMDFA solution is rank-deficient, hence, we have the following corollary.

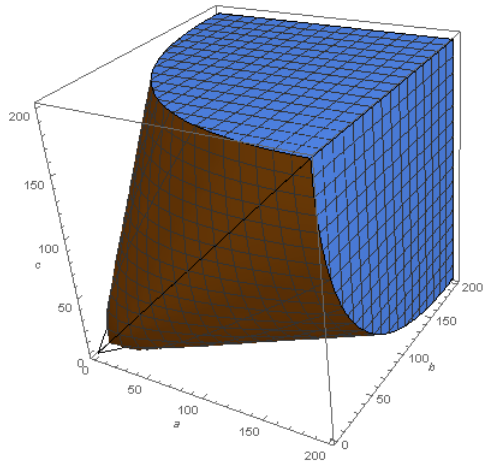


FIGURE 4.2: The feasible region for SNR values s_i which can vary from zero to infinity

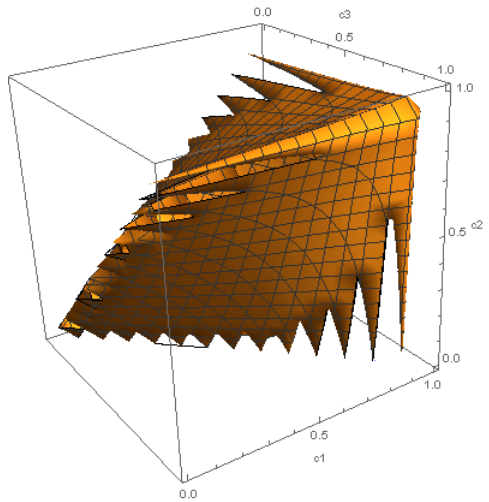


FIGURE 4.3: The feasible region for positive edge weight values a_1, a_2, and a_3 which can vary from zero to one

Corollary 5. *For $n = 3$, and a rank one $\Sigma - D$ following the affine model in* (4.5)*, the CMDFA solution $\Sigma - D^*$ is rank two if and only if all s_i's lie on S_1^c, i.e., the complement of S_1.*

Remark 11. In [63] the lower bound on the rank of $\Sigma - D$ is shown to be the total number of eigenvalues $\lambda_i \in \Sigma$ that are greater than 1. Supposing the 2×2 principal submatrix of Σ, its eigenvalues are $\lambda'_1 = 1 - |\rho_{12}|$ and $\lambda'_2 = 1 + |\rho_{12}|$, where such eigenvalues *interlace* [64] the eigenvalues $\lambda_i,\ i \in \{1, 2, 3\}$ of Σ and we have the ordering $\lambda_1 \leq \lambda'_1 \leq \lambda_2 \leq \lambda'_2 \leq \lambda_3$. The lower bound on λ_2 can be made tighter by considering the intrinsic symmetry in Σ structure, hence having a lower bound $\max 1 - |\rho_{ij}|,\ i \neq j$, i.e., $1 - \min |\rho_{12}|, |\rho_{13}|, |\rho_{23}| \leq \lambda_2$. Now, considering the set S_1, we may see that if one of the SNR values s_i dominates the other two, then (by ignoring the other two SNR values) we may not satisfy the inequality in (4.7). This may happen for example when a single edge-weight a_i is large, i.e., when there is one small correlation value ρ_{ij}, which also will dominate the lower bound on λ_2. As such correlation value decreases, the lower bound becomes closer to 1. Roughly speaking, in this case the search space of rank one matrices $\Sigma - D$ for CMDFA either shrinks or disappears. This might be one reason on having rank two solution on S_1^c in this case.

4.3.4 Non-Star: Rank Two Matrix

Suppose this time that in the affine model the matrix $\Sigma - D$ is a rank two matrix, i.e., $\mathbf{Y} = \{Y_1, Y_2\}$. Hence, the row space of such matrix is two dimensional.

In the following Theorem, whose proof can be found in Appendix 4.5.4, we characterize the solution space of such matrices for rank two CMDFA solutions.

Theorem 8. *For $n = 3$, and a rank two $\Sigma - D$ following the affine model in* (4.5)*, the CMDFA solution is also a rank two matrix if and only if the following system of non-linear equations holds,*

$$\begin{cases} d_3 = 1 + \dfrac{(d_2-1)\rho_{13}^2 + 2\rho_{12}\rho_{13}\rho_{23} + (d_1-1)\rho_{23}^2}{(1-d_1)(1-d_2) - \rho_{12}^2} \\ \dfrac{d_3}{d_1} = \alpha^2 \\ \dfrac{d_3}{d_2} = \beta^2 \end{cases} \tag{4.8}$$

where the parameters α, β are functions of (d_1, d_2) as follows,

$$\begin{aligned} \alpha &= \frac{\rho_{12}\rho_{23} - \rho_{13}(1-d_2)}{(1-d_1)(1-d_2) - \rho_{12}^2} \\ \beta &= \frac{\rho_{12}\rho_{13} - \rho_{23}(1-d_1)}{(1-d_1)(1-d_2) - \rho_{12}^2} \end{aligned} \tag{4.9}$$

Similarly, we have the following Corollary,

Corollary 6. *For $n = 3$, and a rank two $\Sigma - D$ following the affine model in* (4.5)*, the CMDFA solution $\Sigma - D$ is rank one if and only if all d_i's lie on S_2^c, where S_2 is the solution set obtained from Theorem 8.*

Remark 12. The results in Corollaries 5 and 6 interestingly show the difference between the affine models and CMDFA solutions. We may see that regardless of the rank of generative model, i.e., dimension of latent $\mathbf{Y}$ vector, the CMDFA solution can have either lower or higher dimensions. This shows that, in many situations and depending on the values of the transition matrix A, the generative affine model might not be the optimal one (in terms of achieving minimum number of common randomness) to use in order to generate the Gaussian output vector $\mathbf{X}$. Such generative models are usually learned by a specific learning algorithm. For example, for Gaussian latent trees there are efficient algorithms such as Chow-Liu Recursive Grouping (CLRG) [2] and Neighbor Joining [44] algorithm that can consistently learn both the tree structure and parameters.

Remark 13. While CMDFA problem is similar to CMTFA problem with different cost functions (and accepting zero solutions for d_i's), note that their solution sets are different and for the special

case of $n = 3$, they are exclusive. To show this, suppose for CMTFA solution all d_i^*'s are non-zero, then all we need to do is to replace the left hand side of (4.4) with $[1, 1, 1]'$. Essentially, this is because in CMTFA the objective function to be minimized is negative of $Trace(\Sigma - D) = \sum_{i=1}^{3}(1 - d_i)$. Hence, its gradient with respect to $\mathbf{d}$ becomes $[-1, -1, -1]'$. Now, in CMDFA this means replacing the left hand side of (4.4) to $1/\mathbf{d}^* = [1, 1, 1]'$, i.e., $d_1^* = d_2^* = d_3^* = 1$. But then $\Sigma - D^*$ obtains the following form:

$$\Sigma - D^* = \begin{pmatrix} 0 & \rho_{12} & \rho_{13} \\ \rho_{12} & 0 & \rho_{23} \\ \rho_{13} & \rho_{23} & 0 \end{pmatrix} \tag{4.10}$$

In other words $Tr(\Sigma - D^*) = 0$, now if the eigenvalues of $\Sigma - D^*$ are non-zero, then they should have different signs, hence, making $\Sigma - D^*$ non-positive definite, and not Gramian. If they are all zero, then this matrix has a rank zero, and again it violates the Gramian assumption. Note that we assume in CMDFA, the solutions d^* are non-zero, but this cannot be the answer to CMTFA (since then for equivalence of the solutions, we should set all $d_i^* = 1$). Hence, in this case we conclude that the answers for CMTFA and CMDFA are exclusive.

4.3.5 Markov Chain

In [34–36] we showed an operational approach under which any latent Gaussian tree can be efficiently synthesized. We showed that the sources of common randomness can be shrinked into a set of latent variables $\mathbf{X} = \{X_1, ..., X_k\}$ forming a Markov chain structure $X_1 - X_2 - ... - X_k$. Here, we want to show that such chain structure can be efficiently synthesized by a smaller set of variables through an affine Gaussian model. In Theorem 9, whose proof can be found in Appendix 4.5.5 we show these results.

Theorem 9. *Supposing a Gaussian vector $\mathbf{X} \in \mathbb{R}^n$, with Σ following a Markov chain structure $X_1 - X_2 - ... - X_n$, the CMDFA solution $\Sigma - D^*$ has rank either $n - 1$ or $n - 2$.*

In other words, we can always reduce the number of common random bits required to synthesize chain structure through a latent common variables $\{Y_1, ..., Y_k\}$, where $k \in \{n-2, n-1\}$. This also shows that comparing to affine models inducing star structures, such Markov chain structures cannot be significantly made simpler through sum of lower rank and diagonal matrices.

4.4 Conclusion

In this chapter, we studied the problem of characterizing Wyner's common information for Gaussian vectors following special structures, such as star or a Markov chain. We showed that how such problem can be turned into a specific convex programming problem, which we coined as CMDFA. For a general star Gaussian tree, we obtained the linear system of equations that can be efficiently solved to find the CMDFA solution. For $n = 3$ and star Gaussian tree, we showed that interestingly the CMDFA solution can be a rank two matrix. This resulted in computing the general solution space for such case, in which it consists previous solutions as a special case. Finally, for a Gaussian Markov chain we showed that unlike star affine models, these vectors cannot be made as compact such that the lower rank matrix $\Sigma - D$ can be made at most having rank $n - 2$, which suggests that there is not much degree of freedom left to further reduce the model complexity for a Gaussian chain structure.

4.5 Proof of Theorems

4.5.1 Proof of Theorem 5

First, we show the convexity of CMDFA problem. We know that the negative log-determinant function is convex. Moreover, the constraint $D \succ 0$ can be written as n linear constraints of the form $d_i > 0,\ i \in [1, n]$, where d_i's are non-zero diagonal entries of D. The constraint $\Sigma - D \succeq 0$ is equivalent to having its non-negative minimum eigenvalue $\lambda(\mathbf{d})$, or equivalently, $-\lambda(\mathbf{d}) \leq 0$. It is proven in [59, Lemma 1] that in fact finding the negative of minimum eigenvalue corresponds to maximizing a set of linear functions, hence making $-\lambda_{min}(\mathbf{d})$ a convex (piecewise linear)function. So, overall CMDFA is a convex optimization problem.

By relying on [59, Theorem 2] (KKT necessary and sufficient conditions) we obtain that $\mathbf{d}^*$ is a solution to the CMDFA problem defined in (4.3), if and only if $\lambda(\mathbf{d}^*) = 0$, $d_i > 0,\ i \in [1, n]$ and we have the following (with $\alpha \geq 0$)

$$0 \in \nabla(-\log|D|)|_{D=D^*} + \alpha\partial(-\lambda(\mathbf{d}))|_{\mathbf{d}=\mathbf{d}^*} \tag{4.11}$$

Since all d_i's are non-zero (positive), the gradient $\nabla(-\log|D|)|_{D=D^*}$ can be easily replaced as $\dfrac{1}{\mathbf{d}^*} = [1/d_1, ..., 1/d_n]'$. Note that the minimum eigenvalue function is piecewise linear, hence, non-smooth. Since such function is maximum over linear functions, it is shown in [59, Lemma 1] that the subdifferential term can be written as $\partial(-\lambda(\mathbf{d})) = co\bar{n}v\{\mathbf{x}^2\}$, where $co\bar{n}v\{S\}$ denotes the convex hull of set S and the vectors $\mathbf{x}$ are unit eigenvectors of $\Sigma - D$ corresponding to $\lambda(\mathbf{d})$. The term $\mathbf{x}^2$ is the Hadamard product of the vector $\mathbf{x}$ with itself. However, note that as the solution is on the boundary $\lambda(\mathbf{d}^*) = 0$, hence, such eigenvectors in fact correspond to the null space, $N(\Sigma - D)$. Therefore, we rewrite (4.11) as,

$$0 = -\frac{1}{\mathbf{d}^*} + \Sigma_{i=1}^{m}\mathbf{x}_i^2 \tag{4.12}$$

where $\mathbf{x}_i \in N(\Sigma - D)$. Note that m can be arbitrary with $m(m+1)/2 < n$ [58].

Due to [58, Lemma 3.1] one can replace the summation $\Sigma_{i=1}^{m}\mathbf{x}_i^2$ in (4.12) with weighted sum of basis vectors $\mathbf{e}_i \in N(\Sigma - D),\ i \in [1, k]$, with the form $\Sigma_{i=1}^{k}\Sigma_{j=1}^{k} t_{ij}\mathbf{e}_i\mathbf{e}_j^*$. This is due to the fact that any vector $\mathbf{x} \in N(\Sigma - D)$ can be also written as a linear combination of the basis vectors $\mathbf{e}_i \in N(\Sigma - D)$, i.e., there exists $(n-k) \times m$ matrix C such that $[\mathbf{x}_1, \mathbf{x_2}, ..., \mathbf{x}_m] = [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_{n-k}]C$. Then, the summation in (4.12) can be replaced with $\Sigma_{i=1}^{k}\Sigma_{j=1}^{k} t_{ij}\mathbf{e}_i\mathbf{e}_j^*$, where $T = [t_{ij}] = CC'$ and hence it has to be Gramian. Hence, we have the desired results in Theorem 5.

4.5.2 Proof of Theorem 6

Similar to [59, Theorem 4], we may assume that the CMDFA solution $\mathbf{d}_1$ is not unique. Hence, there is another solution $\mathbf{d}_2$, where both of them lie on the boundary $\lambda(\mathbf{d}_1) = \lambda(\mathbf{d}_1) = 0$. From the convexity of the problem, which is shown in Appendix 4.5.1, the vector $\mathbf{d}_3 = (\mathbf{d}_1 + \mathbf{d}_2)/2$

should also be the solution. By [59, Corollary 1] there should be $\mathbf{x} \in N(\Sigma - D_3)$ with non-zero coordinates. We know $N(\Sigma - D_3)\mathbf{x} = 0$. Hence, replacing $\mathbf{d}_3 = (\mathbf{d}_1 + \mathbf{d}_2)/2$, then we obtain $\mathbf{x}'(\Sigma - D_1)\mathbf{x} + \mathbf{x}'(\Sigma - D_2)\mathbf{x} = 0$. Both terms in this summation are non-negative, since the matrices $\Sigma - D_i,\ i \in [1, 2]$ are Gramian. Therefore, we conclude that $\mathbf{x}'(\Sigma - D_1)\mathbf{x} = \mathbf{x}'(\Sigma - D_2)\mathbf{x} = 0$, and we can obtain $(\Sigma - D_1)\mathbf{x} = (\Sigma - D_2)\mathbf{x} = 0$. In particular $D_1\mathbf{x} = D_2\mathbf{x} = 0$, and since D_1 and D_2 are diagonal we may have the system of equations $d_{i1}x_i = d_{i2}x_i, i \in [1, n]$. Since all the vector $\mathbf{x}$ has non-zero entries, hence the uniqueness condition for CMDFA solutions $D_1 = D_2$ holds.

4.5.3 Proof of Theorem 7

Let us find the conditions under which the CMDFA solution is a star. In this case, we have $d_i = 1 - a_i^2$ as the CMDFA solutions. By the results in Section 4.3.1, the null-space basis are $e_1 = (1, 0, -a_3/a_1)$ and $e_2 = (0, 1, -a_3/a_2)$. From (4.4), for this case we obtain the system of equations, $(1/d_1, 1/d_2, 1/d_3) = t_{11}e_1^2 + t_{22}e_2^2 + t_{12}e_1e_2$ that needs to be satisfied for some $T = [t_{ij}] \succeq 0$. solving such system of equations gives us $t_{ii} = \dfrac{a_i^2}{1 - a_i^2},\ i \in [1, 2]$, and $t_{12} = 1/2(s_3 - s_1 - s_2)$, where $s_i = \dfrac{a_i^2}{1 - a_i^2}$. We need to show $T = [t_{ij}] \succeq 0$. Obviously $Trace(T) \geq 0$. For determinant to be non-negative, we need to have $t_{11}t_{22} \geq t_{12}^2$, or $A^2 + B^2 + C^2 - 2AB - 2AC - 2BC \leq 0$. Note that this is not always true, based on the values of s_i, which are the functions of a_i^2. For example, put $s_1 = s_2 = 1$ and $s_3 = 5$, which corresponds to $a_1^2 = a_2^2 = 1/2$ and $a_3^2 = 5/6$ (hence, a positive definite matrix Σ), where one may check that the inequality is not satisfied.

4.5.4 Proof of Theorem 8

Since the row space of $\Sigma - D$ is two-dimensional, we can find non-zero variables α and β such that $\alpha r_1 + \beta r_2 + r_3 = [0, 0, 0]'$, where r_i is the i-th row of $\Sigma - D$. This, of course is a necessary condition for a rank two matrix, and for sufficiency we need to make sure no r_i and r_j are linearly dependent, since otherwise $\Sigma - D$ becomes a rank one matrix.

By replacing r_i's with their respective vectors, we obtain the following system of equations:

$$\begin{aligned}
\alpha(1-d_1)+\beta\rho_{12}+\rho_{13}&=0\\
\alpha\rho_{12}+\beta(1-d_2)+\rho_{23}&=0\\
\alpha\rho_{13}+\beta\rho_{23}+(1-d_3)&=0
\end{aligned} \tag{4.13}$$

Solving for α, β and d_3, gives us:

$$\begin{aligned}
\alpha &= \frac{\rho_{12}\rho_{23}-\rho_{13}(1-d_2)}{(1-d_1)(1-d_2)-\rho_{12}^2}\\
\beta &= \frac{\rho_{12}\rho_{13}-\rho_{23}(1-d_1)}{(1-d_1)(1-d_2)-\rho_{12}^2}\\
d_3 &= 1+\frac{(d_2-1)\rho_{13}^2+2\rho_{12}\rho_{13}\rho_{23}+(d_1-1)\rho_{23}^2}{(1-d_1)(1-d_2)-\rho_{12}^2}
\end{aligned} \tag{4.14}$$

Hence, d_3 can be completely determined, via d_1 and d_2.

We know that the null space $N(\Sigma - D)$ is rank one. And a basis vector can be obtained by solving $(\Sigma - D)\mathbf{x} = \mathbf{0}$. After solving, one deduce that the null space has the following form $N(\Sigma - D) = \{(\alpha, \beta, 1)^T x_3 : \forall x_3\}$, where it turns out that $p = \alpha$ and $q = \beta$. Hence, the basis is $v = (p, q, 1)$, and the normal basis is $e_1 = \dfrac{v}{||v||}$

Now, using Theorem 5 we need to satisfy the following equality $te_1^2 = (1/d_1 1/d_2 1/d_3)^T$ for $t \geq 0$. Which gives us the system of equations $[1/d_1, 1/d_2, 1/d_3]' = \dfrac{1}{||v||^2}[t\alpha^2, t\beta^2, t]'$

Replacing the last equality in the first two, gives us $\dfrac{d_3}{d_1} = \alpha^2$ and $\dfrac{d_3}{d_2} = \beta^2$.

4.5.5 Proof of Theorem 9

First, note that since CMDFA solution is rank-deficient so the rank of solution is at most $n - 1$. Hence we only need to prove that the rank cannot be less than $n-2$. The proof goes by induction. For the bases case, we may consider the case described in Example 8, where we showed the rank of $\Sigma - D$ cannot be less than two.

We show the matrix $\Sigma'_n = \Sigma_n - D$, with Σ_n corresponding to a Markov chain $X_1 - X_2 - .... - X_{n-1} - X_n$, has rank at least $n - 2$; assuming for all $\Sigma'_{n-1} = \Sigma_{n-1} - D$, with Σ_{n-1} regarding

to smaller Markov chains $X_{i_1} - X_{i_2} - ... - X_{i_{n-1}}$ for $i_1 \neq ... \neq i_{n-1} \in [1, n-1]$ have ranks at least $n-3$. In other words, if we sum out (drop) a variable $X_i \in [1, n]$ from the Markov chain, we obtain a length $n-1$ Markov chain with Σ_{n-1}, with rank of Σ'_{n-1} at least least $n-3$.

Without loss of generality, we may assume the Gramian matrix Σ'_n has the following generic form,

$$\Sigma'_n = \begin{pmatrix} & & & \rho_{1,n} \\ & \Sigma'_{n-1} & & \rho_{2,n} \\ & & & \vdots \\ & & & \rho_{n-1,n} \\ \rho_{1,n} & \rho_{2,n} & \cdots & \rho_{n-1,n} & t_n \end{pmatrix} \tag{4.15}$$

Obviously if rank of Σ'_{n-1} is at least $n-2$, then we are done, since the first $n-1$ rows, at least span a $n-2$ dimensional space (adding a new dimension, i.e., the last column, does not reduce the row space dimension). Therefore, we assume Σ'_{n-1} has rank $n-3$.

Consider the first $n-3$ (linearly independent) rows $r_1, ..., r_{n-3}$, and form a linear combination of these rows with row r_n: $\alpha_1 r_1 + ... + \alpha_{n-3} r_{n-3} + \alpha_n r_n$. We are interested to see whether r_n can be written as linear combination of the first $n-3$ rows, and note that $\alpha_n \neq 0$ (since then we have a contradictory conclusion of linear dependence of first $n-3$ rows). Hence, we may ignore α_n and write $\alpha_1 r_1 + ... + \alpha_{n-3} r_{n-3} = r_n$. Extracting the summation elements for the last three columns gives us the following equations,

$$\begin{aligned} \sum_{i=1}^{n-3} \alpha_i \rho_{i,n-2} &= \rho_{n-2,n} \\ \sum_{i=1}^{n-3} \alpha_i \rho_{i,n-1} &= \rho_{n-1,n} \\ \sum_{i=1}^{n-3} \alpha_i \rho_{i,n} &= t_n \end{aligned} \tag{4.16}$$

Due to Markov chain property, we know $\rho_{i,j} = \prod_{(k,l) \in path(i,j)} \rho_{k,l}$, i.e., the pairwise correlation $\rho_{i,j}$ can be computed as the product of all $\rho_{k,l}$, where (x_k, x_l) pairs are the edges on the path

between x_i and x_j. Now, we may multiply the first and second equations by $\rho_{n-2.n}$ and $\rho_{n-1,n}$ and re-write the equations as follows,

$$\begin{aligned}
\sum_{i=1}^{n-3} \alpha_i \rho_{i,n} &= \rho_{n-2,n}^2 \\
\sum_{i=1}^{n-3} \alpha_i \rho_{i,n} &= \rho_{n-1,n}^2 \\
\sum_{i=1}^{n-3} \alpha_i \rho_{i,n} &= t_n
\end{aligned} \tag{4.17}$$

The left hand side on all equations is equal, hence we have $\rho_{n-2,n}^2 = \rho_{n-1,n}^2$, which reduces to $\rho_{n-2,n-1}^2 = 1$, i.e., a rank-deficient Markov chain with rank $n-1$, a contradiction (since we started with a rank n Markov chain).

This shows the linear independence of r_n with first $n-3$ rows, i.e., the set of vectors $(r_1, ..., r_{n-3}, r_n)$ spans an $n-2$ dimensional space, i.e., the rank of Σ_n' is at least $n-2$.

Chapter 5

3D Image Reconstruction Using Factor Models

3D Image reconstruction and hole filling is a classical and fundamentally challenging geometric modeling problem. Such problem happens due to inherent scanning occlusions, hardware noise, and calibration errors during digitization of physical objects, such as 3D scanning of human body.

In this research, we aim to reconstruct and synthesize the missing parts by relying on Gaussian graphical models. We believe that such statistical approach outperforms the deterministic methods.

5.1 Introduction

In this chapter, we consider addressing the hole filling problem for a dataset of 3D scanned human arms. The arm database consists of 1517 sample vectors, each of which consisting 758 3D data point samples, hence making each vector having $758 \times 3 = 2274$ dimension. This problem suffers from *curse of dimensionality* since the number of training samples is much lower than the number of features. Hence, applying factor models to such problem could be helpful.

There are many *deterministic* approaches on image reconstruction aiming to address the image synthesis problem. They can be divided into two categories; *context-insensitive* methods, and *context-based* methods.

• In context-insensitive methods without regarding the underlying template, the missing parts of the image are reconstructed. These methods can be divided into two groups: In the *surface-based* methods [65–67], a cost function is optimized to maximize (minimize) the smoothness (curvature variance) near the missing regions. Essentially, such methods fill the holes by perturbing the input only slightly, but they often cannot handle those special cases, with large curvature variances. In *volume-based* methods [68–70], the object is considered as a solid model, and the missing regions

should be filled. However, similar to the surface-methods they cannot handle complex holes and due to global reconstruction, they lose subtle details in the reconstructed parts.

• Context-based methods usually outperform the context-insensitive methods, since they exploit extra information about the model that can be either a symmetry, given template, or statistical dependency structure of the data points on the image. In *symmetry-guided* methods [71, 72], the inherent symmetry of the object is used to fill the missing parts. However, this only works when the object is symmetric and the missing region's symmetric counterpart is available. In *template-guided* methods [73, 74], the missing regions are obtained and replaced from the set of available templates. However, there might be several occasions when a similar template is unavailable. In *statistical shape models* (SSM) methods, the statistical dependency of the data points in the object is obtained, to further help the image reconstruction problem. Perhaps the best well-known work in SSM field is done by *Cootes, et al.* [75], where a set of *principal component analysis* (PCA) is extracted from the image data points to capture the few statistical variations in the object. Although, this work does not exactly address the hole filling problem, but it is one of the inspiring works in this field.

Factor analysis (FA) is widely recognized as one of the efficient tools to deal with high dimensional problems. In FA, one aims to find the set of latent features that best describes the high dimensional data. The factor models become effective whenever the number of latent variables (features) is much less than the overall number of observed variables. The criterion to find such variables is to optimize a particular cost function. One of the widely used factor models, is *maximum-likelihood* factor analysis (MLFA) model, where the objective is to maximize the (log) likelihood $L(\theta, \mathbb{D})$, where θ is an unknown vector of parameters to be estimated and $\mathbb{D} = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$ is the provided training dataset.

There are other factor models with distinct cost functions. For example the set of works by Shapiro et al. [58–60] defines the trace of a particular covariance matrix as a cost function, which needs to be minimized. They properly characterize the *constrained minimum trace factor analysis*

(CMTFA), and propose an iterative method to solve such *convex programming* problem. Usually, there is no closed form solution for factor models, and one must rely on different optimization techniques to approximate the answer.

5.2 Maximum Likelihood Factor Analysis (MLFA)

We can interpret any 3D object as a random vector with desired joint statistics, where its elements can be anything desirable, such as pixel intensities, or data points locations. Then the hole filling problem can be seen as proposing a synthesis algorithm that generates a random vector whose statistics are very close to the desired object's statistics. In previous sections, we introduced the inherent complexity in high dimensional problems, and how factor models would be helpful in addressing such problems. Although in this chapter, the complexity is more due to *high dimensionality* of data, hence, the applied factor model aims at reducing such complexity by introducing smaller number of latent factors that accurately describe data.

Similar to CMDFA, as discussed in Chapter 4, in MLFA we use an affine model to describe the data. As before, we adopt a joint Gaussian distribution to capture the joint density of data points. We may write a generative model $\mathbf{x}_j = A\mathbf{y}_j + \mu + \mathbf{e}_j$, where $\mathbf{y}_j \sim N(0, I)$ and $\mathbf{e}_j \sim N(0, D)$, with D being diagonal. Hence having the factorization to lower rank and diagonal matrices, $\Sigma = AA' + D$. As discussed, the objective function in MLFA is the log-likelihood function, where it can be written as follows,

$$l(\mu, A, D|\mathbb{D}) = -\frac{n}{2}(ln|\Sigma| + tr(\Sigma^{-1}S) + (\bar{\mathbf{x}} - \mu)'\Sigma^{-1}(\bar{\mathbf{x}} - \mu)) \tag{5.1}$$

where S is the empirical covariance matrix, i.e., $S = 1/n \sum_{j=1}^{n}(\mathbf{x}_j - \bar{\mathbf{x}})'(\mathbf{x}_j - \bar{\mathbf{x}})$. Obviously, the Maximum Likelihood (ML) estimator for μ is $\bar{\mathbf{x}}$. Hence, we can rewrite the above equation as follows,

$$l(A, D|\mathbb{D}) = -\frac{n}{2}(\log|\Sigma| + tr(\Sigma^{-1}S)) \tag{5.2}$$

Taking derivative of $l(A, D|\mathbb{D})$ with respect to both A and D and simplifying the equations gives us the following system of equations,

$$A = S\Sigma^{-1}A$$
$$D = diag(S - AA') \tag{5.3}$$

Solving such equations simultaneously is a difficult problem. The Expectation-Maximization (EM) algorithm [76, 77] is the proposed algorithm to iteratively solve such ML estimation problem. Roughly speaking, EM starts with an initial guess for unknown parameters, computes the likelihood bases on random initial parameters and then maximizes the *complete* log-likelihood to compute the next set of parameters to use for the next iteration.

5.2.1 Expectation Maximization Algorithm

The following steps are taken from [78]. Suppose we are having an initial estimation (A_0, D_0, μ_0) of unknown parameters. We first need to form the following conditional distribution,

$$\mathbf{y}_j|\mathbf{x}_j \sim N(A_0\Sigma^{-1}(\mathbf{x}_j - \mu_0), (I + A_0'D_0^{-1}A_0)^{-1}) \tag{5.4}$$

In particular, the first and second moments of $\mathbf{y}_j$ (under the above conditional distribution) can be computed as follows,

$$E[\mathbf{y}_j|A_0, D_0, \mu_0] = A_0'\Sigma^{-1}(\mathbf{x}_j - \mu_0)$$
$$E[\mathbf{y}_j\mathbf{y}_j'|A_0, D_0, \mu_0] = (I + A_0'D_0^{-1}A_0)^{-1} + A_0'\Sigma^{-1}(\mathbf{x}_j - \mu_0)(\mathbf{x}_j - \mu_0)'\Sigma^{-1}A_0 \tag{5.5}$$

Now, we may form the *complete* data log-likelihood over all vectors in the dataset $\mathbb{D}$ as follows,

$$l(A, D, \mu) = \sum_{j=1}^{n} \log p(\mathbf{x}_j, \mathbf{y}_j) = \sum_{j=1}^{n} \log p(\mathbf{y}_j) + \log p(\mathbf{x}_j|\mathbf{y}_j)$$
$$= -\frac{n}{2}ln|D| - \frac{1}{2}\sum_{j=1}^{n}(\mathbf{y}_j'\mathbf{y}_j + (\mathbf{x}_j - A\mathbf{y}_j - \mu)'D^{-1}(\mathbf{x}_j - A\mathbf{y}_j - \mu)) \tag{5.6}$$

Now, we need to compute the expected likelihood $l(A, D, \mu)$ with respect to the conditional distribution $p(\mathbf{y}_j|\mathbf{x}_j)$ defined above,

$$E[l(A, D, \mu)|A_0, D_0, \mu_0] = Q(A, D|A_0, D_0) + R \tag{5.7}$$

where,

$$Q(A, D|A_0, D_0) = -\frac{n}{2} ln|D| \\ -\frac{n}{2} tr[D^{-1}S - 2D^{-1}A_0A_0'\Sigma_0^{-1}S + (I + A'D^{-1}A)((I + A_0'D_0^{-1}A_0)^{-1} + A_0'\Sigma_0^{-1}S\Sigma_0^{-1}A_0)] \tag{5.8}$$

and,

$$-2R/n = (\bar{\mathbf{x}} - \mu_0)'\Sigma_0^{-1}A_0(I + A'D^{-1}A)A_0'\Sigma_0^{-1}(\bar{\mathbf{x}} - \mu_0) \\ + (\bar{\mathbf{x}} - \mu)'D^{-1}(\bar{\mathbf{x}} - \mu) - 2(\bar{\mathbf{x}} - \mu)'D^{-1}AA_0'\Sigma_0^{-1}(\bar{\mathbf{x}} - \mu_0) \tag{5.9}$$

Note that if we set $\mu_0 = \bar{\mathbf{x}}$, then R is maximized at $\mu = \bar{\mathbf{x}}$. Hence, at all iterations we use such ML estimator, i.e., $\mu^t = \bar{\mathbf{x}}$. Next, we can take the derivative of $Q(A, D|A_0, D_0)$ and show that it is maximized for,

$$A = SD_0^{-1}A_0[I + A_0'\Sigma_0^{-1}SD_0^{-1}A_0]^{-1} \\ D = diag(S - S\Sigma_0^{-1}A_0A') \tag{5.10}$$

Hence, at each step of EM algorithm we have,

$$A_{t+1} = SD_t^{-1}A_t[I + A_t'\Sigma_t^{-1}SD_t^{-1}A_t]^{-1} \\ D_{t+1} = diag(S - S\Sigma_t^{-1}A_tA_{t+1}') \tag{5.11}$$

Note that $\Sigma_t = A_tA_t' + D_t$.

5.3 Statistical Analysis on Arm Dataset

5.3.1 Eigenvalue Spectrum Analysis

Let's begin by observing the eigenvalue spectrum of the empirical covariance matrix, S These are shown in Figure 5.1.

The maximum eigenvalue is $\lambda_1 \approx 850$, while the smallest one is $\lambda_n \approx 0$. As it may seem from the figure, huge number of eigenvalues are around zero. This shows that the determinant is almost

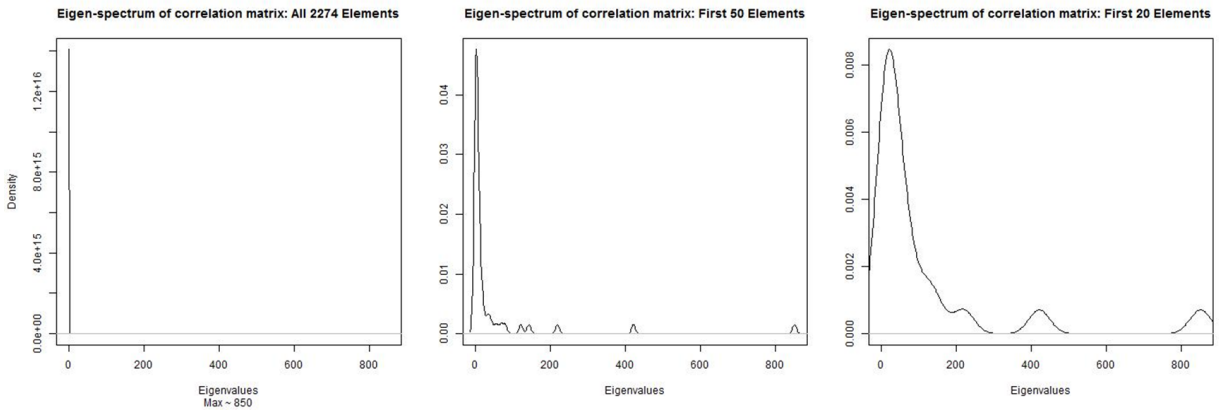


FIGURE 5.1: Eigenvalues spectrum of S: Left: All eigenvalues; Middle: first 50 eigenvalues; Right: first 20 eigenvalues

zero, since S can be computed as a product of all 2274 eigenvalues. One possible explanation is that to describe an arm pose, one do not need 2274 features. Hence, by introducing 2274 features, we are simply many redundant variables, i.e., those that are linearly dependent to the other variables, hence, they (almost) will be determined once we observe the value of other variables.

5.4 Comparing Different Methods

We compare MLFA with several other statistical learning methods. Except the *Naive approaches* that we will discuss next, all other approaches adopt the conditional mean of missing variables, given the observed variables and underlying conditions and structures, as a predicted value for the missing variables. Hence, as it will be shown, their underlying conditions (seen as hyper-parameters for these models) have a significant impact on their prediction performance.

5.4.1 Naive Approaches

We show the performance for two naive approaches. The prediction error is defined to be the expected L_2 norm between the predicted vector $\hat{\mathbf{x}}$ and the original vector $\mathbf{x}$. First, we simply use the mean vector μ_{ML} obtained from training set to do the deterministic prediction. Then, we use the empirical distribution $N(\Sigma_{ML}, \mu_{ML})$ to generate several independent arm samples (without conditioning on any observation).

As we will see, using only mean vector gives us better performance, while the random sampling is more prone to producing very large errors.

5.4.2 Chow-Liu (CL) Tree

We also use Chow-Liu (CL) tree [79] to approximate the joint density of all 2274 random variables. The optimization criterion in finding such tree is to minimize the KL distance between the empirical and CL distributions. This is shown to be equivalent to finding a maximum weight spanning tree, where weights are defined to be the pairwise mutual information values between variables [79]. We first need to estimate the edge-weights in Chow-Liu tree, then we may use the learned structure to predict the missing data points. Hence, this way we are actually using the observed data to predict the hidden variables. For example, Figure 5.2 shows the estimated CL tree structure for each of the coordinates of 3D data points in the dataset. Although it is hard to recognize, but all of such graphical models are trees with Gaussian joint density. The CL trees we used for estimation purposes consists of all the coordinates (variables) in the dataset.

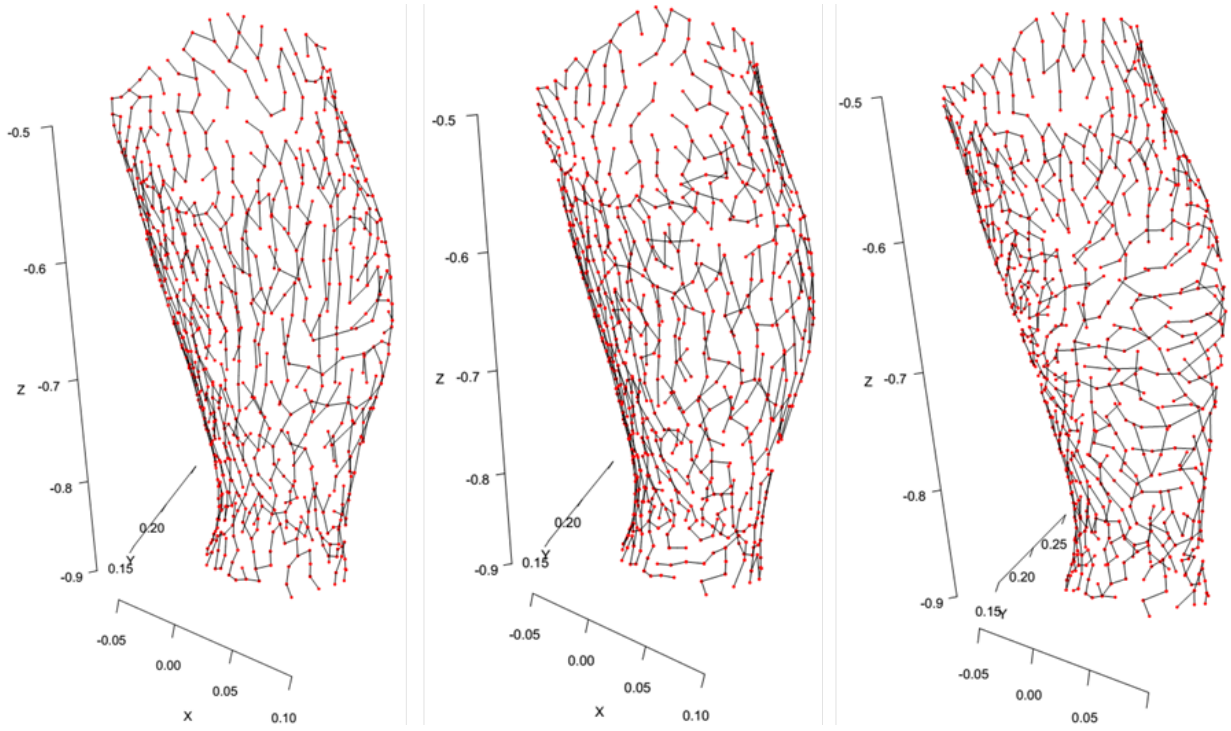


FIGURE 5.2: Estimated CL trees for each coordinate x, y, z of the 3D data points

In predicting the missing data points we use a *synchronous* Gibbs sampling technique. This iterative approach begins by initializing all the missing values, by randomly drawing a sample from an empirical joint distribution. Then, at each iteration we update all the missing values, by drawing a sample from a conditional distribution. Figure 5.3 shows three steps of this algorithm.

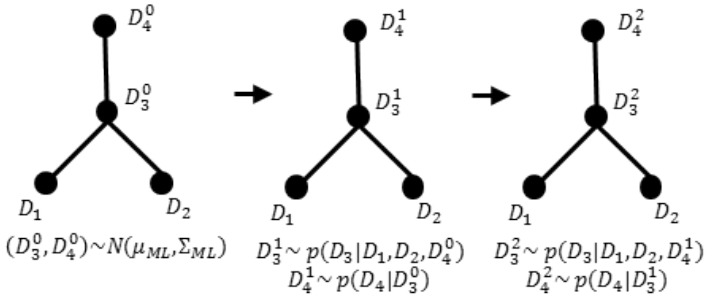


FIGURE 5.3: Synchronous Gibbs Sampling: First, we initialize the missing data points D_3 and D_4 by randomly sampling from their joint empirical distribution. At the next step, we update the conditional distributions of each missing variable by considering all of their observed neighbors along with hidden neighbors' samples from previous slots.

5.4.3 MLFA with Gibbs Sampling

In previous sections, we showed how to use EM algorithm to estimate both $A \in R^{n \times k}$ and $D \in R^{n \times n}$ in an affine factor model. Now, supposing that we learned both A and D parameters we are ready to use such a model as our predictor. Based on the observed data and similar to discussions for CL tree we implement a *synchronous Gibbs sampling* method to produce reasonable samples for hidden variables. Roughly speaking, we first initialize all missing variables using their joint prior, i.e., joint Gaussian density. Then, we use all such values $\mathbf{X}_o \cup \mathbf{X}_m$ (i.e., the set of observed and missing variables, respectively) to update the conditional density of all latent factors $\{Y_1, Y_2, ..., Y_k\}$. We randomly sample from the obtained conditional density and use them to update the conditional density of missing variables $\mathbf{X}_m$, and to update the missing variables by sampling from those updated conditional densities. This goes back and forth, until we meet some predefined threshold.

Note that the joint covariance matrix of all the nodes $\mathbf{Y} \cup \mathbf{X}$ in this case has the following form:

$$\Sigma_{\mathbf{Y},\mathbf{X}} = \begin{pmatrix} I & A' \\ A & AA' + D \end{pmatrix} \tag{5.12}$$

Hence, the latent variables are updated using the conditional distributions, as follows:

$$\begin{aligned} \Sigma_{\mathbf{Y}|\mathbf{X}} &= I - A^T(AA^T + D)^{-1}A \\ \mu^{(i)}_{\mathbf{Y}|\mathbf{X}} &= \mu_{\mathbf{Y}} + A^T(AA^T + D)^{-1}(\mathbf{x_i} - \mu_{ML}) \end{aligned} \tag{5.13}$$

Finally, the missing values in $\mathbf{X}_m$ can be updated as follows:

$$\begin{aligned} \Sigma_{\mathbf{X}|\mathbf{Y}} &= (AA^T + D) - AIA^T = D \\ \mu^{(i+1)}_{\mathbf{X}|\mathbf{Y}} &= \mu_{\mathbf{X}} + A(\mathbf{y_i} - \mu_{\mathbf{Y}}) \end{aligned} \tag{5.14}$$

Note that in (5.14) we do not apply all the obtained values to the vector $\mathbf{X}$. We only update the missing values $\mathbf{X}_m \subset \mathbf{X}$.

5.4.4 Maximum Likelihood MLFA

Due to specific closed form solutions obtained from Gaussian assumption, here, even without using Gibbs sampling, one can rely on FA parameters to estimate the missing data points. Note that this *one step* method should reach the same performance as Gibbs sampler, since essentially both estimators aim at maximizing the likelihood function, given the observed variables under the same model. This is done by observing that now, we modeled Σ as the sum $AA' + D$. Hence, given the observed and missing variables, we can break Σ into four blocks: the missing variables covariance matrix Σ_m, the observed-missing variables covariance matrix Σ_{om} and Σ_{mo}; and the observed variables covariance matrix Σ_o. Now, we can perform the Maximum-Likelihood estimation for missing variables, based on observed data, which would be the same as finding the conditional mean vector regarding to missing variables:

$$\mu_{m|o} = \mu_m + \Sigma_{mo}\Sigma_{oo}^{-1}(\mathbf{x}_o - \mu_o) \tag{5.15}$$

Note that such approach results in only one particular output, and is not well-suited for generative modeling applications, where we need to synthesize set of distinct, but reasonable outputs.

5.4.5 Maximum Likelihood PPCA

Probabilistic Principal Component Analysis (PPCA) [80] is performed under the same conditions as in MLFA, with a difference that in PPCA we restrict the noise elements to be homogeneous, i.e., having an identical variance values. Hence, from the model $\Sigma = AA' + D$, the diagonal matrix D should be replaced by $\sigma^2 I$, where σ^2 is a variable to be optimized and $I_{n\times n}$ is the identity matrix.

5.4.6 Graphical Lasso

Another way of constraining the estimated model is to explicitly add a regularization term in order to penalize dense learned structures. Roughly speaking, we force the optimization to find more sparse solutions rather than dense overfitted models. To this end, the least absolute shrinkage and selection operator (LASSO) model introduced by Tibshirani in [81]. Such method is further extended to graphical models, and a Graphical Lasso (GLasso) is defined in [82]. The overall format of the problem can be formulated as follows:

$$\hat{\Theta} = arg\max_{\Theta \succ 0} \{\mathbf{log}|\mathbf{\Theta}| - \mathbf{trace}(\mathbf{S\Theta}) - \lambda||\mathbf{\Theta}||_1\} \tag{5.16}$$

where $\Theta = \Sigma^{-1}$ is the precision matrix, and $||.||_1$ is the L_1 norm operator on a given matrix. Here, using different values of λ we penalize each element θ_{ij} in the precision matrix.

The equation (5.16) should be represented in a more simple way in order to become analytically solvable. The authors in [83] used a Black Coordinate Descent algorithm by defining the *dual* problem in order to iteratively solve (5.16). We would expect by introducing such constraint we capture the sparsity in data and by avoiding the overfitting, we may increase the prediction accuracy.

5.5 Experimental Setup

5.5.1 Choosing the Cardinality of Latent Factors

Finding necessary number of latent factors, i.e., k is a design issue. Here, we consider enough number of factors to capture more than 99% of all 2274 variance of variables. Hence, we begin with the first latent factor Y_1 and its corresponding eigenvalue λ_1 (the largest eigenvalue), then we check whether the fraction $\frac{\lambda_1}{\sum_{i\in 1}^{2274}\sigma_i^2}$ satisfies the threshold, otherwise, we keep adding the eigenvalues, and recall that sum of all eigenvalues is basically equal to the sum of all variances till we reach the threshold. For this dataset the threshold is satisfied after including the first 50 eigenvalues, hence 50 latent factors are needed.

5.5.2 L_2 Performance Comparison

In comparing the L_2 errors of each model, we applied them to two different scenarios: Randomly missing variables and missing chunk of variables. In particular, we first randomly choose between 500 and 1000 variables out of 2274 variables and hide their values from the test dataset. In the second case, we search for the highest variance (with most uncertainty) chunk of data with size 500 and 1000 and drop the regarding variables from the test dataset. These variables form the set $\mathbf{X}_m$ while the remaining ones form the observed set $\mathbf{X}_o$. The results are shown in Table 5.1.

TABLE 5.1: Normalized L_2 : $E[\frac{\sqrt{\sum_{i=1}^{M}(E_i)^2}}{M}]$ (in cm), error comparison, with M as number of missing variables and $E[.]$ taken over all test-arm images (≈ 517)

		Random		Chunk	
		500	1000	500	1000
Method	Mean	2.42e-2	1.71e-2	2.96e-2	1.85e-2
	Random	2.50e-2	1.78e-2	3.07e-2	1.91e-2
	ML-PPCA	4.12e-3	3.06e-3	7.46e-3	9.07e-3
	GLASSO ($\lambda = .05$)	3.18e-3	2.64e-3	1.04e-2	1.41e-2
	ML-CL	2.97e-3	2.51e-3	1.56e-2	1.94e-2
	ML-MLFA	2.56e-3	1.84e-3	5.1e-3	4.74e-3
	Gibbs-MLFA	2.59e-3	1.86e-3	5.09e-3	6.57e-3

As it can be seen that through all cases the MLFA model outperforms all other estimators. In particular, the performance of Gibbs-MLFA and ML-MLFA is similar, and it is expected to become identical as we increase the number of iterations in Gibbs sampling. Although ML-CL has competitive performance for random missing variables case with MLFA, but its performance drops significantly in high variance chunks of missing variables. This might be due to the fact that CL tree uses neighborhood observed points to recover the missing point. But since a chunk of missing points leads to very few observed neighbors around the boundary of missing chunk, the performance in such case drops significantly. This may also be the reason on decreased accuracy of GLasso method when chunks of variables are missing, since similar to CL, in GLasso we are seeking for locally sparse structures. Finally, as we expected, the performance of ML-PPCA is worse than ML-MLFA, due to the fact that PPCA model has homogeneity condition on additive noise variances that results in a more restricted optimization problem compared to MLFA.

5.6 Conclusion

In this section, we studied the application of factor models in model selection and estimation problems, and in particular 3D image reconstruction problem. We showed that for 3D arm dataset, a particular factor model, i.e., MLFA can outperform all other statistical estimators bya significant margin. We showed that due to robust nature of MLFA, i.e., considering a handful of latent factors to describe the overall joint density of data, we've been able to provide high accuracy even for extreme cases when many missing values have high variance and uncertainty.

Chapter 6

Summary and Future Works

The goal of this research is to study the topological and algebraic properties f Gaussian graphical models. These models are proved to be beneficial in modeling stochastic complex systems. Our research consisted of several parts that started with privacy and security applications in Chapter 2, efficient synthesis of such graphical models in Chapters 3 and 4, and finally to show applications of such models in image reconstruction applications in Chapter 5. Below, we give a summary of our contributions and provide several interesting future research ideas for each study.

6.1 Extractable Common Randomness from Gaussian Trees: Topological and Algebraic Perspectives

In Chapter 2, we study both topological and algebraic properties of unrooted Gaussian trees and characterized their security performance. Such performance is measured by the corresponding potential in extracting common randomness from a given tree, which is further determined by max-min and min-max conditional mutual information values, subject to the order of selecting variables from the tree by legitimate nodes Alice and Bob, and an eavesdropper Eve, respectively. A new operation is proposed to transform a Gaussian tree into another, and also to order different Gaussian trees. Through such operation we construct several equivalent classes of Gaussian trees. Each class includes multiple Gaussian trees that can be partially ordered based on the associated max-min or min-max conditional mutual information (CMI) metric, and thus we can find the most secure and the least secure trees in each partially ordered set (poset). The union of all posets generates all possible non-isomorphic trees of the given number of variables. Then, we assign a particular polynomial to each Gaussian tree, and show that such polynomial can determine the relative security performance of the Gaussian tree with respect to other trees within the same class. In the end, based on a generalized integer partition method, we propose a novel approach

to efficiently enumerate the most secure structures of all posets. One may consider the following directions for future studies:

• One direction is to generalize such idea not only to Gaussian trees but to any general Gaussian graphical model structure. This results in more complex correlation situations, where the degree of association is not only a function of a unique route between two variables.

• Another interesting direction is to consider random vectors instead of random variables. In particular, what if Alice, Bob, and Eve can choose several variables to increase/decrease their communication secrecy. A new metric should be define to define such secrecy, and both max-min and min-max problems become more complex in a sense that not only their inter-relations would impact the secrecy but their intra-relations inside each user would affect such metric as well.

6.2 Layered Synthesis of Latent Gaussian Trees

In Chapter 3, a new synthesis scheme is proposed to generate a random vector with prescribed joint density that induces a (latent) Gaussian tree structure. The quality of synthesis is shown by vanishing total variation distance between the synthesized and desired statistics. The proposed layered and successive synthesis scheme relies on the learned structure of tree to use sufficient number of common random variables to synthesize the desired density. We characterize the achievable rate region for the rate tuples of multi-layer latent Gaussian tree, through which the number of bits needed to synthesize such Gaussian joint density are determined. The random sources used in our algorithm are the latent variables at the top layer of tree, the additive independent Gaussian noises, and the Bernoulli sign inputs that capture the ambiguity of correlation signs between the variables. We have shown that such ambiguity can further help in reducing the synthesis rates for the underlying Gaussian trees. Some interesting future ideas are as follows,

• Providing efficient synthesis approach for any Gaussian graphical mode structure. Although such approach heavily relies on finding the optimal rate through solving CMDFA problem introduced in Chapter 4.

6.3 Algebraic Properties of Solutions to Common Information of Gaussian Graphical Models

In Chapter 4, we formulate Wyner's common information for random vectors $\mathbf{x} \in \mathbb{R}^n$ with joint Gaussian density. We show that finding the common information of Gaussian vectors is equivalent to maximizing a log-determinant of the additive Gaussian noise covariance matrix. We coin such optimization problem as a constrained minimum determinant factor analysis (CMDFA) problem. The convexity of such problem with necessary and sufficient conditions on CMDFA solution is shown. We study the algebraic properties of CMDFA solution space, through which we study two extreme Gaussian graphical models, namely, *latent Gaussian stars*, and *explicit Gaussian chains*. Interestingly, we show that depending on pairwise covariance values in a Gaussian graphical structure, one may not always end up with the same parameters and structure for CMDFA solution as those found via graphical learning algorithms. Several further studies can be taken as follows,

- We only studied the solution space for CMDFA problem for star Gaussian trees with three nodes. It would be interesting to generalize such idea to star Gaussian trees with more than three nodes, and study the changes in solution space. In other words, whether as number of variables increases the solution space of CMDFA for star Gaussian trees vanishes.

- We provided a necessary and sufficient conditions under which a matrix D^* is a solution to CMDFA. The effect of Null space in a more practical and intuitive way remains an open problem for future researches. In particular, it would be interesting to see how the change in null space, either the dimension or the basis vectors affects the CMDFA solution.

6.4 3D Image Reconstruction Using Factor Models

In Chapter 5, we studied a particular 3D image reconstruction problem. 3D Image reconstruction and hole filling is a classical and fundamentally challenging geometric modeling problem. Such problem happens due to inherent scanning occlusions, hardware noise, and calibration errors during digitization of physical objects, such as 3D scanning of human body.

In Chapter 5, we showed how to reconstruct and synthesize the missing parts by relying on Gaussian graphical models. And through experimental results showed that such statistical approach outperforms the deterministic methods. Several future directions for such study can be taken,

• Despite their popularity, the latent features in factor models do not convey any intuitive meaning and they are simply mathematical functions derived from observed features. It would be interesting to see how each factor is related to a physical feature of image. There are some recent works [84] that aim to interpret the meaning of such latent factors by assigning new observed features similar to them that can helpful for such studies.

• We can interpret any 3D object as a random vector with desired joint statistics, where its elements can be anything desirable, such as pixel intensities, or data points locations. Then the hole filling problem can be seen as proposing a synthesis algorithm that generates a random vector whose statistics are very close to the desired object's statistics. In Chapter 4, we found that there is fundamental similarities between common information and CMTFA problem from factor analysis. In fact, we showed that in common information problem, we are dealing with *constrained minimum determinant factor analysis* (CMDFA). Hence, similar tools from factor analysis would be beneficial in solving CMDFA as well. Similar steps may be taken to show the relations of MLFA with CMDFA, and how each approach would be beneficial in certain settings. Such study explains the possible relations between the models chosen by common information and factor analysis, where as of now, each of them can be seen as pure optimization problems. Addressing such questions will be helpful to come up with possibly a hybrid image synthesis algorithm which inherits the efficiency from both fields.

References

[1] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Foundations and Trends® in Machine Learning*, vol. 1, no. 1-2, pp. 1–305, 2008.

[2] M. J. Choi, V. Y. Tan, A. Anandkumar, and A. S. Willsky, "Learning latent tree graphical models," *The Journal of Machine Learning Research*, vol. 12, pp. 1771–1812, 2011.

[3] E. Abbe and C. Sandon, "Community detection in general stochastic block models: Fundamental limits and efficient algorithms for recovery," in *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*. IEEE, 2015, pp. 670–688.

[4] ——, "Recovering communities in the general stochastic block model without knowing the parameters," in *Advances in Neural Information Processing Systems*, 2015, pp. 676–684.

[5] I. Cabreros, E. Abbe, and A. Tsirigos, "Detecting community structures in hi-c genomic data," in *2016 Annual Conference on Information Science and Systems (CISS)*. IEEE, 2016, pp. 584–589.

[6] E. Abbe and M. J. Wainwright. [Online]. Available: http://www.princeton.edu/eabbe/publications/tuto-slides-part1.pdf

[7] ——. [Online]. Available: http://www.princeton.edu/eabbe/publications/tuto-slides-part2.pdf

[8] A. A. Amini and M. J. Wainwright, "High-dimensional analysis of semidefinite relaxations for sparse principal components," in *2008 IEEE International Symposium on Information Theory*. IEEE, 2008, pp. 2454–2458.

[9] N. P. Santhanam and M. J. Wainwright, "Information-theoretic limits of selecting binary graphical models in high dimensions," *IEEE Transactions on Information Theory*, vol. 58, no. 7, pp. 4117–4134, 2012.

[10] U. M. Maurer, "Secret key agreement by public discussion from common information," *Information Theory, IEEE Transactions on*, vol. 39, no. 3, pp. 733–742, May 1993.

[11] R. Ahlswede and I. Csiszár, "Common randomness in information theory and cryptography. part i: secret sharing," *IEEE Transactions on Information Theory*, vol. 39, no. 4, July 1993.

[12] ——, "Common randomness in information theory and cryptography. II. cr capacity," *Information Theory, IEEE Transactions on*, vol. 44, no. 1, pp. 225–240, Jan. 1998.

[13] S. Chaudhary and G. Gordon, "Tutte polynomials for trees," *Journal of graph theory*, vol. 15, no. 3, pp. 317–331, July 1991.

[14] A. D. Wyner, "The common information of two dependent random variables," *Information Theory, IEEE Transactions on*, vol. 21, no. 2, pp. 163–179, 1975.

[15] T. S. Han and S. Verdú, "Approximation theory of output statistics," *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 752–772, 1993.

[16] P. Cuff, "Distributed channel synthesis," *Information Theory, IEEE Transactions on*, vol. 59, no. 11, pp. 7071–7096, 2013.

[17] A. Moharrer, S. Wei, G. T. Amariucai, and J. Deng, "Evaluation of security robustness against information leakage in Gaussian polytree graphical models," in *2015 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2015, pp. 1404–1409.

[18] ——, "Topological and algebraic properties for classifying unrooted Gaussian trees under privacy constraints," in *2015 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2015, pp. 1–6.

[19] A. Moharrer, S. Wei, G. Amariucai, and J. Deng, "Extractable common randomness from Gaussian trees: Topological and algebraic perspectives," *Transactions on Information Forensics and Security, IEEE*, vol. 10, no. 11, pp. 2312–2321, 2016.

[20] M. H. Manshaei, Q. Zhu, T. Alpcan, T. Bacşar, and J.-P. Hubaux, "Game theory meets network security and privacy," *ACM Computing Surveys (CSUR)*, vol. 45, no. 3, p. 25, June 2013.

[21] S. Chaudhuri, "Qualitative inequalities for squared partial correlations of a Gaussian random vector," *Annals of the Institute of Statistical Mathematics*, vol. 66, no. 2, pp. 345–367, Apr. 2014.

[22] G. E. Andrews, *The theory of partitions*. Cambridge university press, 1998, vol. 2.

[23] A. Moharrer, S. Wei, G. Amariucai, and J. Deng, "Evaluation of security robustness against information leakage in Gaussian polytree graphical models," *Wireless Communications and Networking Conference (WCNC), IEEE*, pp. 1404–1409, Mar. 2015.

[24] ——, "Topological and algebraic properties for classifying unrooted Gaussian trees under privacy constraints," *To appear in proceedings of the Global Communications Conference (GlobeCom)*, Dec. 2015.

[25] S. Sullivant, "Algebraic geometry of Gaussian Bayesian networks," *Advances in Applied Mathematics*, vol. 40, no. 4, pp. 482–513, May 2008.

[26] H. Roozbehani and Y. Polyanskiy, "Algebraic methods of classifying directed graphical models," *arXiv preprint arXiv:1401.5551*, Jan. 2014.

[27] P. Šimecek, "Gaussian representation of independence models over four random variables," in *COMPSTAT conference*, 2006.

[28] K. Patra and A. Lal, "The effect on the algebraic connectivity of a tree by grafting or collapsing of edges," *Linear Algebra and its Applications*, vol. 428, no. 4, pp. 855–864, Feb. 2008.

[29] P. Csikvári, "On a poset of trees II," *Journal of Graph Theory*, vol. 74, no. 1, pp. 81–103, Sept. 2013.

[30] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.

[31] A. Moharrer, S. Wei, G. Amariucai, and J. Deng, "Extractable common randomness from gaussian trees: Topological and algebraic perspectives," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 10, pp. 2312–2321, 2016.

[32] W. T. Trotter, *Combinatorics and partially ordered sets: Dimension theory*. JHU Press, 2001, vol. 6.

[33] D. Eisenstat and G. Gordon, "Non-isomorphic caterpillars with identical subtree data," *Discrete mathematics*, vol. 306, no. 8, pp. 827–830, May 2006.

[34] A. Moharrer, S. Wei, G. T. Amariucai, and J. Deng, "Synthesis of gaussian trees with correlation sign ambiguity: An information theoretic approach," in *Communication, Control, and Computing (Allerton), 2016 54th Annual Allerton Conference on*. IEEE, 2016, pp. 378–384.

[35] ——, "Layered synthesis of latent Gaussian trees," in *Submitted to IEEE Transactions of Information Theory*. IEEE, 2017.

[36] ——, "Successive synthesis of latent gaussian trees," in *Accepted to MILCOM 2017, 2017 IEEE Military Communications conference*. IEEE, Oct. 2017.

[37] T. D. Kulkarni, P. Kohli, J. B. Tenenbaum, and V. Mansinghka, "Picture: A probabilistic programming language for scene perception," in *Proceedings of the ieee conference on computer vision and pattern recognition*, 2015, pp. 4390–4399.

[38] C. Gourieroux, A. Monfort, and E. Renault, "Indirect inference," *Journal of applied econometrics*, vol. 8, no. S1, pp. S85–S118, 1993.

[39] E. Cameron and A. Pettitt, "Approximate bayesian computation for astronomical model analysis: a case study in galaxy demographics and morphological transformation at high redshift," *Monthly Notices of the Royal Astronomical Society*, vol. 425, no. 1, pp. 44–65, 2012.

[40] Y. Jin and S. Geman, "Context and hierarchy in a probabilistic image model," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2. IEEE, 2006, pp. 2145–2152.

[41] A. Dobra, T. S. Eicher, and A. Lenkoski, "Modeling uncertainty in macroeconomic growth determinants using gaussian graphical models," *Statistical Methodology*, vol. 7, no. 3, pp. 292–306, 2010.

[42] R. Mourad, C. Sinoquet, N. L. Zhang, T. Liu, P. Leray, *et al.*, "A survey on latent tree models and applications." *J. Artif. Intell. Res.(JAIR)*, vol. 47, pp. 157–203, 2013.

[43] N. Shiers, P. Zwiernik, J. A. Aston, and J. Q. Smith, "The correlation space of gaussian latent tree models," *arXiv preprint arXiv:1508.00436*, 2015.

[44] N. Saitou and M. Nei, "The neighbor-joining method: a new method for reconstructing phylogenetic trees." *Molecular biology and evolution*, vol. 4, no. 4, pp. 406–425, 1987.

[45] G. V. Steeg, S. Gao, K. Reing, and A. Galstyan, "Sifting common information from many variables," *arXiv preprint arXiv:1606.02307*, 2016.

[46] G. Xu and B. Chen, "Information for inference," in *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*. IEEE, 2011, pp. 1516–1520.

[47] P. Yang and B. Chen, "Wyner's common information in gaussian channels," in *IEEE International Symposium on Information Theory (ISIT)*, 2014, pp. 3112–3116.

[48] G. Xu, W. Liu, and B. Chen, "A lossy source coding interpretation of wyner's common information," *Information Theory, IEEE Transactions on*, vol. 62, no. 2, pp. 754–768, 2016.

[49] Q. Chen, F. Cheng, T. Liu, and R. W. Yeung, "A marginal characterization of entropy functions for conditional mutually independent random variables (with application to wyner's common information)," in *IEEE International Symposium on Information Theory (ISIT)*, 2015, pp. 974–978.

[50] G. J. Op't Veld and M. C. Gastpar, "Caching gaussians: Minimizing total correlation on the gray-wyner network," in *Information Science and Systems (CISS), 2016 Annual Conference on*. IEEE, 2016, pp. 478–483.

[51] S. Satpathy and P. Cuff, "Gaussian secure source coding and wyner's common information," *arXiv preprint arXiv:1506.00193*, 2015.

[52] M. C. Gastpar *et al.*, "Total correlation of gaussian vector sources on the gray–wyner network," in *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, no. EPFL-CONF-222848, 2016.

[53] A. Moharrer, S. Wei, G. T. Amariucai, and J. Deng, "Synthesis of Gaussian Trees with Correlation Sign Ambiguity: An Information Theoretic Approach," *arXiv preprint*, 2016. [Online]. Available: "http://arxiv.org/abs/1601.06403"

[54] Y. Wu and S. Verdu, "The impact of constellation cardinality on gaussian channel capacity," in *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*. IEEE, 2010, pp. 620–628.

[55] B. Li, S. Wei, Y. Wang, and J. Yuan, "Chernoff information of bottleneck gaussian trees," in *IEEE International Symposium on Information Theory (ISIT)*, July 2016.

[56] A. Moharrer and S. Wei, "Algebraic properties of solutions to common information of Gaussian graphical models," in *Submited to 55th Allerton Conference*. IEEE, 2017.

[57] L. Vandenberghe, S. Boyd, and S.-P. Wu, "Determinant maximization with linear matrix inequality constraints," *SIAM journal on matrix analysis and applications*, vol. 19, no. 2, pp. 499–533, 1998.

[58] A. Shapiro, "Rank-reducibility of a symmetric matrix and sampling theory of minimum trace factor analysis," *Psychometrika*, vol. 47, no. 2, pp. 187–199, 1982.

[59] G. Della Riccia and A. Shapiro, "Minimum rank and minimum trace of covariance matrices," *Psychometrika*, vol. 47, no. 4, pp. 443–448, 1982.

[60] A. Shapiro, "Weighted minimum trace factor analysis," *Psychometrika*, vol. 47, no. 3, pp. 243–264, 1982.

[61] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[62] ——, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, ser. Lecture Notes in Control and Information Sciences, V. Blondel, S. Boyd, and H. Kimura, Eds. Springer-Verlag Limited, 2008, pp. 95–110, http://stanford.edu/~boyd/graph_dcp.html.

[63] L. Guttman, "Some necessary conditions for common-factor analysis," *Psychometrika*, vol. 19, no. 2, pp. 149–161, 1954.

[64] S. Fisk, "A very short proof of cauchy's interlace theorem for eigenvalues of hermitian matrices," *arXiv preprint math/0502408*, 2005.

[65] P. Liepa, "Filling holes in meshes," in *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*. Eurographics Association, 2003, pp. 200–205.

[66] J.-P. Pernot, G. Moraru, and P. Véron, "Filling holes in meshes using a mechanical model to simulate the curvature variation minimization," *Computers & Graphics*, vol. 30, no. 6, pp. 892–902, 2006.

[67] C. Xiao, W. Zheng, Y. Miao, Y. Zhao, and Q. Peng, "A unified method for appearance and geometry completion of point set surfaces," *The Visual Computer*, vol. 23, no. 6, pp. 433–443, 2007.

[68] T. Ju, "Robust repair of polygonal models," *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 888–895, 2004.

[69] J. Davis, S. R. Marschner, M. Garr, and M. Levoy, "Filling holes in complex surfaces using volumetric diffusion," in *3D Data Processing Visualization and Transmission, 2002. Proceedings. First International Symposium on*. IEEE, 2002, pp. 428–441.

[70] S. Bischoff and L. Kobbelt, "Structure preserving cad model repair," in *Computer Graphics Forum*, vol. 24, no. 3. Wiley Online Library, 2005, pp. 527–536.

[71] A. Sharf, M. Alexa, and D. Cohen-Or, "Context-based surface completion," *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 878–887, 2004.

[72] M. Pauly, N. J. Mitra, J. Giesen, M. H. Gross, and L. J. Guibas, "Example-based 3d scan completion," in *Symposium on Geometry Processing*, no. EPFL-CONF-149337, 2005, pp. 23–32.

[73] V. Kraevoy and A. Sheffer, "Template-based mesh completion." in *Symposium on Geometry Processing*, vol. 385, 2005, pp. 13–22.

[74] R. Gal, A. Shamir, T. Hassner, M. Pauly, and D. Cohen-Or, "Surface reconstruction using local shape priors," in *Symposium on Geometry Processing*, no. EPFL-CONF-149318, 2007, pp. 253–262.

[75] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, "Active shape models-their training and application," *Computer vision and image understanding*, vol. 61, no. 1, pp. 38–59, 1995.

[76] Factor analysis and kalman filtering. [Online]. Available: https://people.eecs.berkeley.edu/~jordan/courses/281A-fall04/lectures/lec-11-2.pdf

[77] Expectation maximization, fa and pca. [Online]. Available: https://www.cs.princeton.edu/courses/archive/fall10/cos513/notes/2010-11-17.pdf

[78] J.-H. Zhao, P. L. Yu, and Q. Jiang, "Ml estimation for factor analysis: Em or non-em?" *Statistics and computing*, vol. 18, no. 2, pp. 109–123, 2008.

[79] C. Chow and C. Liu, "Approximating discrete probability distributions with dependence trees," *IEEE transactions on Information Theory*, vol. 14, no. 3, pp. 462–467, 1968.

[80] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611–622, 1999.

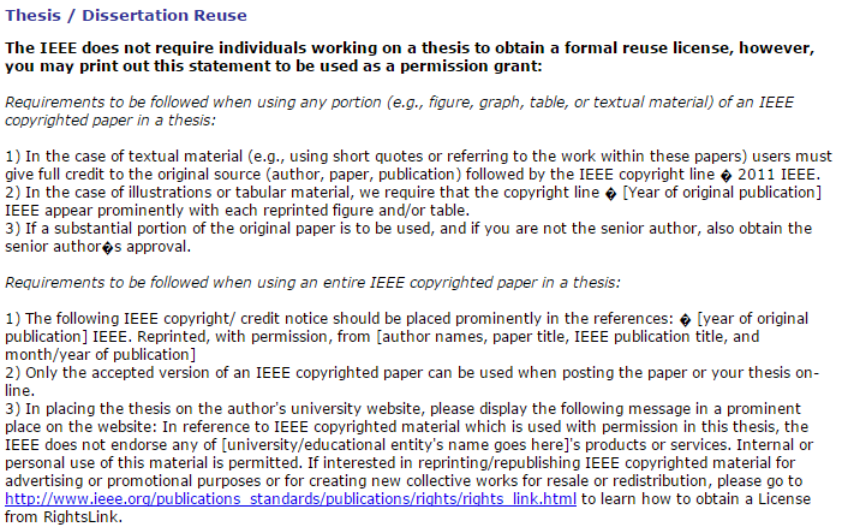
[81] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

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

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

[84] A. Taeb and V. Chandrasekaran, "Interpreting latent variables in factor models via convex optimization," *arXiv preprint arXiv:1601.00389*, 2016.

Appendix: Permission Request



Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

- Permission to reprint from IEEE Title: Moharrer, Ali, Shuangqing Wei, George T. Amariucai, and Jing Deng. "Evaluation of security robustness against information leakage in Gaussian polytree graphical models." In Wireless Communications and Networking Conference (WCNC), 2015 IEEE, pp. 1404-1409. IEEE, 2015., IEEE Thesis / Dissertation Reuse.
- Permission to reprint from IEEE Title: Moharrer, Ali, Shuangqing Wei, George T. Amariucai, and Jing Deng. "Topological and algebraic properties for classifying unrooted Gaussian trees under privacy constraints." In Global Communications Conference (GLOBECOM), 2015 IEEE, pp. 1-6. IEEE, 2015., IEEE Thesis / Dissertation Reuse.
- Permission to reprint from IEEE Title: Moharrer, Ali, Shuangqing Wei, George T. Amariucai, and Jing Deng. "Extractable common randomness from Gaussian trees: Topological and algebraic perspectives." IEEE Transactions on Information Forensics and Security 11, no. 10 (2016): 2312-2321., IEEE Thesis / Dissertation Reuse.
- Permission to reprint from IEEE Title: Moharrer, Ali, Shuangqing Wei, George T. Amariucai, and Jing Deng. "Synthesis of Gaussian trees with correlation sign ambiguity: An information theoretic approach." In Communication, Control, and Computing (Allerton), 2016 54th Annual Allerton Conference, pp. 378-384. IEEE, 2016., IEEE Thesis / Dissertation Reuse.
- Permission to reprint from IEEE Title: Moharrer, Ali, Shuangqing Wei, George T. Amariucai, and Jing Deng. "Successive Synthesis of Latent Gaussian Trees." To Appear In MILCOM 2017. IEEE, 2017., IEEE Thesis / Dissertation Reuse.
- Permission to reprint from IEEE Title: Moharrer, Ali and Shuangqing Wei. "Synthesis of Gaussian trees with correlation sign ambiguity: An information theoretic approach." To Appear In Communication, Control, and Computing (Allerton), 2017 55th Annual Allerton Conference, IEEE, 2017., IEEE Thesis / Dissertation Reuse.

Vita

Ali Moharrer received her B.Sc. in Electrical Engineering from the Zanjan University (ZNU) , Zanjan, Iran in 2010. From 2010 to 2013, he worked as a Research Assistant during his M.Sc. at the Amirkabir University of Technology (AUT), Tehran, Iran. From 2013 to 2017, he studied Electrical Engineering as a PhD student at the school of Electrical Engineering and Computer Science (EECS) at the Louisiana State University (LSU), Baton Rouge, LA. He is also with the Information Sensing, Learning, and Security Laboratory, Louisiana State University. His research interests include information theory and its applications in generative models and deep learning systems.