Determining the Distributed Karhunen-Loève Transform via Convex Semidefinite Relaxation

DETERMINING THE DISTRIBUTED KARHUNEN-LOÈVE TRANSFORM VIA CONVEX SEMIDEFINITE RELAXATION

BY

XIAOYU ZHAO, B.Eng.

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

Master of Applied Science (2018) McMaster University

(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE:	Determining the Distributed Karhunen-Loève Transform via Convex Semidefinite Relaxation
AUTHOR:	Xiaoyu Zhao B.Eng., (Communication Engineering) University of Electronic Science and Technology of China, Chengdu, China
SUPERVISOR:	Dr. Jun Chen
NUMBER OF PAGES:	xi, 53

To my family and the one I love

Abstract

The Karhunen-Loève Transform (KLT) is prevalent nowadays in communication and signal processing. This thesis aims at attaining the KLT in the encoders and achieving the minimum sum rate in the case of Gaussian multiterminal source coding.

In the general multiterminal source coding case, the data collected at the terminals will be compressed in a distributed manner, then communicated the fusion center for reconstruction. The data source is assumed to be a Gaussian random vector in this thesis. We introduce the rate-distortion function to formulate the optimization problem. The rate-distortion function focuses on achieving the minimum encoding sum rate, subject to a given distortion. The main purpose in the thesis is to propose a distributed KLT for encoders to deal with the sampled data and produce the minimum sum rate.

To determine the distributed Karhunen-Loève transform, we propose three kinds of algorithms. The first iterative algorithm is derived directly from the saddle point analysis of the optimization problem. Then we come up with another algorithm by combining the original rate-distortion function with Wyner's common information, and this algorithm still has to be solved in an iterative way. Moreover, we also propose algorithms without iterations. This kind of algorithms will generate the unknown variables from the existing variables and calculate the result directly.

All those algorithms can make the lower-bound and upper-bound of the minimum sum rate converge, for the gap can be reduced to a relatively small range comparing to the value of the upper-bound and lower-bound.

Acknowledgements

I would like to express my gratitude to all the people who have offered me help on my graduate thesis. First and foremost, I would like to thank my supervisor Dr. Jun Chen for his encouragement and instruction. His rigorous attitudes as well as enthusiasms on research has positively influenced me on my life. I'm honored to be a student of Dr. Chen, and it has been an unforgettable and fortunate experience in my academic years.

I am also very grateful to Dr.Tim Davidson and Dr.Jiankang Zhang as my defense committee members. And I would like to thank all the stuff and faculty members in the ECE department at McMaster. What's more, I'd like to thank all my friends, Yameng, Dan, Xuan and etc, for our friendship.

Finally, I would like to thank my family for their generous support and love for me. It is them who inspire me to conquer all the difficulties in my life bravely and confidently.

Notation and abbreviations

$\mathrm{E}[\cdot]$	the expectation operator
$\mathrm{tr}(\cdot)$	the trace operator
$(\cdot)^{\mathrm{T}}$	the transpose operator
$\mathrm{diag}(x_1, \cdots, x_\ell)$	an diagonal matrix with the i-th diagonal entry being $x_i, i = 1, \cdots, \ell$
$\lvert \cdot \rvert$	the determinant operator
$I(X;Y)$	the mutual information between variables X and Y
PCA	principal component analysis
KLT	Karhunen-Loève transform

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

The basic task of source coding is to characterize the signal with the minimum number of symbols, under the constraint of a certain acceptable level of distortion. The techniques of source coding are generally divided into two parts, as lossless coding and lossy coding.

The lossless coding indicates the coding method that can restore the original data exactly from the compressed data. However, lossless coding can still reduce the code rate, by exploiting the contained statistical properties or dependencies in the data compaction. However, there's a limit in the application of lossless coding, for the lossless coding can only be used for the signals of discrete-time and discrete-amplitude [1]. One of the characteristic application of lossless coding is JPEG-LS [2].

Lossy source coding has a more widespread application, including the compression of speech, pictures, video and audio signals, where the complete reconstruction of the original source data is not required. Typical examples of the lossy coding include

JPEG [3] for picture coding and H.264/AVC for video coding [4].

In the case of lossy coding, the restored signal is not completely the same as the original source, but is only an approximation of it. Thus, we take the distortion as a measure for the deviation of the original source and the restored data. Apart from the distortion, there's another property required for evaluating the performance of coding, which is rate. When coding a sequence of finite symbols, the transmission rate is defined as the average number of bits per input symbol [5].

There is a major branch in information theory named rate-distortion theory, which focuses on achieving the minimum encoding rate to reconstruct the original source data, subject to a given distortion. Since the determining of distributed KLT is based on the trade-off relationship between rate and distortion in this thesis, we will introduce the rate-distortion function first as a foundation.

It is clear that when the random variable is continuous, it is impossible for it to be precisely represented with a finite number of bits, and hence the coding process should be lossy. In typical applications, the ultimate receiver of the compressed signal is a human sense, and it is also unnecessary to avoid the loss completely, due to the limited sensitivity of human perception. However, we still try to achieve a better coding result, to make the reconstruction as good as possible. This work can be completed by defining a rate-distortion function, which is treated as a rate-distance between the source and its output after coding. The rate-distortion function is usually referred as $R(d)$, where d is a given value, indicating that the distortion should be equal or less than d.

One basic channel model is presented in Figure 1.1. where $X(1), ..., X(n)$ are independent and identically distributed (i.i.d) process. Define the original source as $X^n =$

$(X(1), \cdots, X(n))$ and the restored data from decoder as $\hat{X^n} = (\hat{X}(1), \cdots, \hat{X}(n))$. The rate-distortion function in the lossy coding case is represented as follows,

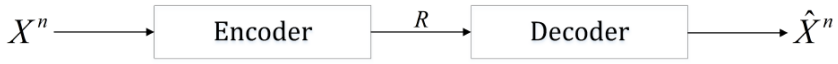

Figure 1.1: The basic channel model

$$\frac{1}{n}\Sigma_{t=1}^{n} E[(X(t) - \hat{X}(t))^2] \leq d \tag{1.1}$$

$$R(d) \triangleq \min_{P_{\hat{X}|X}: E[(X(t)-\hat{X}(t))^2] \leq d} I(X; \hat{X}). \tag{1.2}$$

In the scalar quadratic Gaussian case, $X \sim \mathcal{N}(0, \Sigma_X)$.

$$R(d) = \frac{1}{2} \log^+ (\frac{\Sigma_X}{d}), \tag{1.3}$$

where $\log^+(a) \triangleq \max\{\log a, 0\}$.

1.2 The Karhunen-Loève transform

In this thesis, our interest mainly focus on the distributed Karhunen-Loève transform (KLT). Consider the input source vector $X = (X_1, \cdots, X_L)^T$ in a multiterminal source coding scenario with L terminals, each terminal will sample a part of the input vector X_i, $i = 1, ..., L$. The data sampled at different terminals will be compressed in a distributed manner, then communicated to the fusion center for reconstruction. The core problem in the thesis is to propose a distributed coding method for the encoders to deal with the collected data through the KLT, in order to achieve the

minimum sum rate in the multi-terminal case. By constructing the rate-distortion function and solving the optimal convex problem, the KLT at each terminal can be attained.

Since the approximation and compression of data is prevalent nowadays, the KLT has become a fundamental part in plenty of related cases, which is also considered as a type of principal component analysis (PCA).

In the standard (nondistributed) KLT situation, there exists only one overall encoder, which will collect the whole signal source $X = (X_1, \cdots, X_L)^T$. Here we assume X is in the quadratic Gaussian case, $X \sim \mathcal{N}(0, \Sigma_X)$. The encoder here plays a role to generate the description of the entire vector X in a certain manner, thus the reconstruction part can provide the estimated output vector $\hat{X} = (\hat{X}_1, \cdots, \hat{X}_L)^T$ from the rate-distortion function. Here we put the sum distortion constraint as

$$\frac{1}{n}\sum_{t=1}^{n}\sum_{i=1}^{L} E[(X_i(t) - \hat{X}_i(t))^2] \leq d. \tag{1.4}$$

Then we introduce one variable D as the distortion matrix, which is defined as

$$D = E[(X - \hat{X})(X - \hat{X})^T]. \tag{1.5}$$

And the rate-distortion function in the nondistributed case is represented as

$$\begin{aligned} R(d) &= \min_{D} \frac{1}{2}\log\left(\frac{|\Sigma_X|}{|D|}\right) \\ \text{subject to} \quad & D \preceq \Sigma_X \\ & \text{tr}(D) \leq d. \end{aligned} \tag{1.6}$$

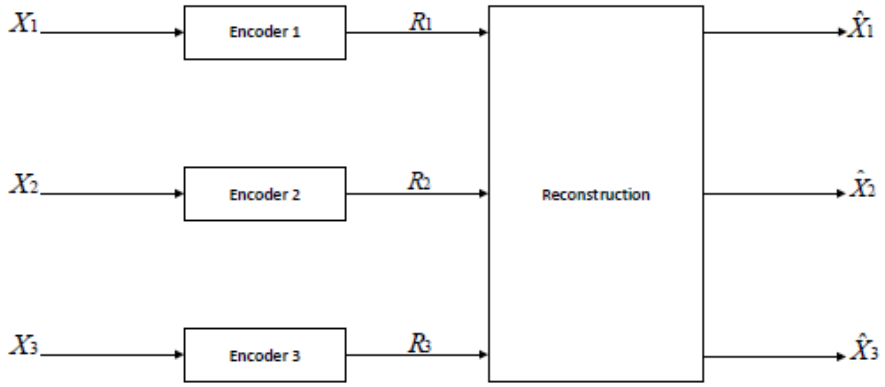

Figure 1.2: The distributed KLT scenario, $L = 3$.

Then we apply KLT on Σ_X and obtain the unitary matrix U and diagonal matrix Λ as follows,

$$\begin{aligned} \Sigma_X &= U^T \Lambda U \\ \Lambda &= \text{diag}(\lambda_1, \cdots, \lambda_L). \end{aligned} \tag{1.7}$$

In this situation, we can apply the reverse water filling algorithm to represent the problem in (1.6) as

$$R(d) = \frac{1}{2} \log^+ \left(\frac{\lambda_i}{\eta} \right), \tag{1.8}$$

where $\Sigma_{i=1}^{L} \min\{\eta, \lambda_i\} = \min(d, \text{tr}(\Sigma_X))$.

Apart from the standard KLT scenario, there are cases where applying KLT on the entire vector is not possible. In such case, the KLT should be employed in a distributed manner.

In the distributed KLT scenario, there exists multiple separate encoders, which is shown in Fig 1.2 (suppose L=3). Since those encoders are not able to communicate with each other in this situation, it is clear that applying the KLT to the entire source

signal vector is not possible here. Instead, each encoder deals with a part of the source signal vector X (which is $X_i, i = 1, ..., L$) separately and produces the corresponding output Y_i at the rate $R_i, i = 1, ..., L$. Here we assume X to be the Gaussian case, $X \sim \mathcal{N}(0, \Sigma_X)$. Similar to the centralized case, the reconstruction part will also produce the estimated vector $\hat{X}$ from the rate-distortion function. In the distributed scenario, we define the distortion d_i in each encoder i as

$$\frac{1}{n}\sum_{t=1}^{n} E[(X_i(t) - \hat{X}_i(t))^2] \leq d_i, \quad i = 1, ..., L. \tag{1.9}$$

Let σ_i denote the error variance of encoder i, $i = 1, ..., L$. The minimum sum rate achievable in the distributed KLT is given by

$$\begin{aligned} &\min_{\sigma_1,\cdots,\sigma_L} \frac{1}{2}\log\left(\frac{|\Sigma_X|}{|D|}\right) \\ &\text{subject to } \ \sigma_i \leq \sigma_{X_i} \\ &D = (\Sigma_X^{-1} + \text{diag}(\sigma_1^{-1} - \sigma_{X_1}^{-1}, \cdots, \sigma_L^{-1} - \sigma_{X_L}^{-1}))^{-1} \\ &\text{tr}(D) \leq d. \end{aligned} \tag{1.10}$$

One fundamental part in the above problem is characterizing the optimum trade-off between the compression rates and distortions [6]. The lossless case of this problem was solved by Slepian and Wolf to a large extent, in the paper published in 1973 [7]. Later, their result has been extended to the lossy case by Wyner and Ziv [8], as well as Tung [9]. Although there has not existed a complete solution yet to the problem of multiterminal source coding in the general case, remarkable development has been made on some special scenarios, especially in the quadratic Gaussian case [10–15].

1.3 Organization of this thesis

In this thesis, we address the multiterminal source coding problem in a distributed Karhunen-Loève transform (KLT) scenario. The topic of this thesis is inspired by the work of Michael Gastpar *et al.* [16], which also determines the architecture of KLT in multiterminal source coding model through the rate-distortion function. This thesis will extend the work of Gastpar *et al.* by relaxing the equality constraint and proposing a convex optimization formulation.

The organization of the rest part is introduced as follows. The problem definitions and conversions will be represented in Chapter 2. The solution algorithms, the simulation results and the compares between them can be found in Chapter 3, Chapter 4 and Chapter 5. we will draw the conclusions in Chapter 6.

Chapter 2

Preliminaries

2.1 Conversions of the problem

In this section, we will make conversions to the original rate-distortion function, in order to transform the original objective problem. The algorithms for the distributed Karhunen-Loève transform(KLT) will be derived from this newly converted convex optimization problem.

In the multiterminal source coding, there are L terminals, and each terminal will sample a part of the input source vector X. In the transform coding case, let X be a Gaussian random vector with mean zero and covariance matrix Σ_X. Define $X^T = (X_1^T, \cdots, X_L^T)^T$, where X_i is a vector of dimension ℓ_i, $i = 1, ..., L$. Let $Y^T = (Y_1^T, \cdots, Y_L^T)^T$ be the output vector after encoding, with $Y_i = C_i X_i + Q_i$, where C_i is a matrix with dimension $\ell_i \times \ell_i$, and Q_i is also a Gaussian random vector with mean zero and covariance Σ_{Q_i}, $i = 1, ..., L$. Here the matrix C_i acts as the transform matrix in each encoder i for realizing the KLT. And the vector Q_i is assumed to be the noise term. Hence Q_i and Q_j $(i \neq j,\ i, j = 1, ..., L)$ are independent with each other.

It is assumed that the vectors X_i and Q_j $(j = 1, ..., L)$ are mutually independent. Further, we define the matrix C from matrix C_i as $C = \text{diag}(C_1, \cdots, C_L)$ and let matrix $\Sigma_Q = \text{diag}(\Sigma_{Q_1}, \cdots, \Sigma_{Q_L})$.

The objective function is derived as follows, from the rate-distortion function mentioned in (1.2). The expectation $E(X|Y)$ can represent the vector $\hat{X}$ mentioned in Chapter 2, which is the data reconstructed from vector Y. Therefore, we can derive the expression for distortion matrix D from (1.5) as

$$D = \text{tr}(E[(X - E[X|Y])(X - E[X|Y])^T]). \tag{2.1}$$

The mutual information $I(X; \hat{X})$ is equals to the mutual information $I(X; Y)$. Therefore, the problem in (1.2) can be converted to

$$\begin{aligned} &\min_{C, \Sigma_Q \succeq 0} I(X; Y) \\ &\text{subject to} \quad \text{tr}(E[(X - E[X|Y])(X - E[X|Y])^T]) \leq d. \end{aligned} \tag{2.2}$$

Because of the conditions where X and Y are both Gaussian distributed and Q_i are mutually independent from X_i, the objective function can also be represented as

$$I(X; Y) = \frac{1}{2} \log \frac{|C\Sigma_X C^T + \Sigma_Q|}{|\Sigma_Q|}. \tag{2.3}$$

The derivation of (2.3) is listed as follows,

$$\begin{aligned} I(X;Y) &= I(Y;X) \\ &= \frac{1}{2}\log\frac{|\Sigma_Y|}{|E[(X-Y)(X-Y)^T]|} \\ &= \frac{1}{2}\log\frac{|C\Sigma_X C^T+\Sigma_Q|}{|\Sigma_Q|}. \end{aligned}$$

As for the distortion matrix D in (2.1), it can also be derived as

$$\begin{aligned} D &= \mathrm{tr}(E[(X-E[X|Y])(X-E[X|Y])^T]) \\ &= \Sigma_X - \Sigma_X C^T(C\Sigma_X C^T+\Sigma_Q)^{-1}C\Sigma_X. \end{aligned} \tag{2.4}$$

Therefore, the optimization problem in (2.2) can be written as

$$\begin{aligned} &\min_C \frac{1}{2}\log\frac{|C\Sigma_X C^T+\Sigma_Q|}{|\Sigma_Q|} \\ &\text{subject to } \ \mathrm{tr}(\Sigma_X - \Sigma_X C^T(C\Sigma_X C^T+\Sigma_Q)^{-1}C\Sigma_X) \le d \\ &\qquad \Sigma_Q \succeq 0. \end{aligned} \tag{2.5}$$

Since the optimization problem has been turned into the form in (2.5), we can introduce several new variables to make the conversions to a further step. Let Σ_i be an $\ell_i \times \ell_i$ matrix, where $i = 1, ..., L$. We assume that

$$0 \preceq \mathrm{diag}(\Sigma_1, \cdots, \Sigma_L) \preceq \Sigma_X. \tag{2.6}$$

Then define another matrix Σ_Z from $\Sigma_i, i = 1, \cdots, L$, and the covariance matrix Σ_X as follows,

$$\Sigma_Z = (\mathrm{diag}(\Sigma_1^{-1}, \cdots, \Sigma_L^{-1}) - \Sigma_X^{-1})^{-1}. \tag{2.7}$$

The case of Σ_Z may appear rank-deficient, but this situation can be handled via a suitable projection to the nondegenerate subspace. Moreover, introduce another new variable matrix $\Gamma_i, i = 1, ..., L$ as

$$\Gamma_i = \Sigma_i - \Sigma_i C_i^T (C_i \Sigma_i C_i^T + \Sigma_{Q_i})^{-1} C_i \Sigma_i. \tag{2.8}$$

There's one equation for describing the relationship between Γ_i, D and Σ_Z, which can be verified as

$$\text{diag}(\Gamma_1, \cdots, \Gamma_L) = (D^{-1} + \Sigma_Z^{-1})^{-1}. \tag{2.9}$$

With all those newly introduced variables including Σ_i, $\Gamma_i (i = 1, ..., L)$, Σ_Z and D, it can be verified that

$$\frac{1}{2} \log \frac{|C\Sigma_X C^T + \Sigma_Q|}{|\Sigma_Q|} = \frac{1}{2} \log \frac{|\Sigma_X + \Sigma_Z||\text{diag}(\Sigma_1, \cdots, \Sigma_L)|}{|D + \Sigma_Z||\text{diag}(\Gamma_1, \cdots, \Gamma_L)|}. \tag{2.10}$$

The derivation of those two conversions (2.9) and (2.10) is given in the appendix. Therefore, we can write the optimization problem in (2.5) to be an optimization problem as

$$\begin{aligned} \min_{D, \Gamma_1, \cdots, \Gamma_L} \ & \frac{1}{2} \log \frac{|\Sigma_X + \Sigma_Z||\text{diag}(\Sigma_1, \cdots, \Sigma_L)|}{|D + \Sigma_Z||\text{diag}(\Gamma_1, \cdots, \Gamma_L)|} \\ \text{subject to} \ \ & 0 \preceq D \preceq \Sigma_X, \\ & \text{tr}(D) \leq d, \\ & 0 \preceq \text{diag}(\Gamma_1, \cdots, \Gamma_L), \\ & \text{diag}(\Gamma_1, \cdots, \Gamma_L) = (D^{-1} + \Sigma_Z^{-1})^{-1}. \end{aligned} \tag{2.11}$$

Here we can relax the last constraint to generate the new following convex optimization problem as

$$\begin{aligned} \min_{D,\Gamma_1,\cdots,\Gamma_L} \ & \frac{1}{2}\log\frac{|\Sigma_X+\Sigma_Z||\mathrm{diag}(\Sigma_1,\cdots,\Sigma_L)|}{|D+\Sigma_Z||\mathrm{diag}(\Gamma_1,\cdots,\Gamma_L)|} \\ \text{subject to} \ \ & 0 \preceq D \preceq \Sigma_X, \\ & \mathrm{tr}(D) \leq d, \\ & 0 \preceq \mathrm{diag}(\Gamma_1,\cdots,\Gamma_L), \\ & \mathrm{diag}(\Gamma_1,\cdots,\Gamma_L) \preceq (D^{-1}+\Sigma_Z^{-1})^{-1}. \end{aligned} \tag{2.12}$$

This newly generated optimization problem provides the lower bound on the original non-convex problem. Note that this lower bound is valid for any $\Sigma_1,\cdots,\Sigma_L$ satisfying (2.6). Therefore, we can maximize over $\Sigma_1,\cdots,\Sigma_L$ subject to (2.6) to find the tightest lower bound.

Chapter 3

Implementation of The Iterative Algorithm

3.1 The saddle point

In the convex optimization problem (2.12), we assume matrix $\Sigma_i, i = 1, ..., L$ to be constant, thus the optimization problem is solved over the variables D and $\Gamma_i, i = 1, ..., L$ to get the lower bound on minimum sum rate. In fact, the matrix Σ_i is also a variable and has influence on the result of this minimum sum rate as well. Therefore, our tightest lower bound on the sum rate is represented by the following max-min

problem,

$$\begin{aligned}
&\max_{\Sigma_1,\cdots,\Sigma_L}\ \min_{D,\Gamma_1,\cdots,\Gamma_L}\ \frac{1}{2}\log\frac{|\Sigma_X+\Sigma_Z||\mathrm{diag}(\Sigma_1,\cdots,\Sigma_L)|}{|D+\Sigma_Z||\mathrm{diag}(\Gamma_1,\cdots,\Gamma_L)|}\\
&\text{subject to}\ \ 0\preceq \mathrm{diag}(\Sigma_1,\cdots,\Sigma_L)\preceq\Sigma_X\\
&\qquad\qquad 0\preceq D\preceq\Sigma_X,\\
&\qquad\qquad \mathrm{tr}(D)\le d,\\
&\qquad\qquad 0\preceq \mathrm{diag}(\Gamma_1,\cdots,\Gamma_L),\\
&\qquad\qquad \mathrm{diag}(\Gamma_1,\cdots,\Gamma_L)\preceq(D^{-1}+\Sigma_Z^{-1})^{-1}.
\end{aligned}\tag{3.1}$$

In the following analysis, we assume the number of terminals L to be 2. In the above optimization problem, $\mathrm{diag}(\Sigma_1,\Sigma_2)=(\Sigma_X^{-1}+\Sigma_Z^{-1})^{-1}$, which is derived from the definition of Σ_Z in (2.7). By applying the matrix inversion lemma, we can get the following equation

$$(D^{-1}+\Sigma_Z^{-1})^{-1}=D-D(\Sigma_Z+D)^{-1}D. \tag{3.2}$$

Now we can rewrite the last constraint as

$$0\preceq D-\mathrm{diag}(\Gamma_1,\Gamma_2)-D(\Sigma_Z+D)^{-1}D,$$

by the Schur complement, this is equivalent to

$$0\preceq\begin{pmatrix}\Sigma_Z+D & D\\ D & D-\mathrm{diag}(\Gamma_1,\Gamma_2)\end{pmatrix}$$

which is linear matrix inequality. Then we'd like to make some conversions on the

objective function in (3.1). Introduce a new variable Θ, where

$$(\Sigma_X^{-1} + \Theta^{-1})^{-1} = D. \tag{3.3}$$

Therefore, the objective function is concave over Σ_1 and Σ_2. It is possible that there exists a saddle point and the max-min problem is equivalent to min-max problem as follows,

$$\begin{aligned} \min_{D,\Gamma_1,\Gamma_2} \max_{\Sigma_1,\Sigma_2} \frac{1}{2} \log \frac{|\Sigma_X + \Theta||\mathrm{diag}(\Sigma_1,\Sigma_2)|}{|\mathrm{diag}(\Sigma_1,\Sigma_2) + \Theta|\mathrm{diag}(\Gamma_1,\Gamma_2)|} \\ \text{subject to } \ 0 \preceq \mathrm{diag}(\Sigma_1,\Sigma_2) \preceq \Sigma_X \\ 0 \preceq D \preceq \Sigma_X, \\ \mathrm{tr}(D) \leq d, \\ 0 \preceq \mathrm{diag}(\Gamma_1,\Gamma_2), \\ (\mathrm{diag}(\Gamma_1^{-1},\Gamma_2^{-1}) - \Theta^{-1})^{-1} \preceq \mathrm{diag}(\Sigma_1,\Sigma_2) \preceq \Sigma_X. \end{aligned} \tag{3.4}$$

3.2 The iterative method

In the solving part of this thesis, we utilize the CVX toolbox and the SDPT3 in MATLAB to crack those optimization problems. Here CVX is a modeling system for convex optimization and SDPT3 is a software for semidefinite-quadratic-linear programming.

When the optimization problem is represented in the form like (3.4) with all these constraints, the optimization problem itself is convex, but CVX won't recognize it to be so. Therefore, we have to transform the problem in (3.4) by linearization and make CVX able to solve it.

Here we generate an iterative algorithm to get the minimum value on sum rate and this algorithm is divided into two parts.

The first min part of the problem deal with the variables D, Γ_1, Γ_2 and treat the matrix Σ_1, Σ_2 as constants, the objective function and constraints for this part is listed as follows,

$$\begin{aligned} \min_{D,\Gamma_1,\Gamma_2} \ & \frac{1}{2}\log\frac{|\Sigma_X+\Sigma_Z||\mathrm{diag}(\Sigma_1,\Sigma_2)|}{|D+\Sigma_Z||\mathrm{diag}(\Gamma_1,\Gamma_2)|} \\ \text{subject to} \ \ & 0 \preceq D \preceq \Sigma_X, \\ & \mathrm{tr}(D) \le d, \\ & 0 \preceq \mathrm{diag}(\Gamma_1,\Gamma_2), \\ & \mathrm{diag}(\Gamma_1,\Gamma_2) \preceq (D^{-1}+\Sigma_Z^{-1})^{-1}. \end{aligned} \tag{3.5}$$

Then we deal with max part of the problem, obtaining the maximum value of the objective function from the variables Σ_1 and Σ_2. Similarly, here we treat the variables D, Γ_1 and Γ_2 as constants, with the corresponding value they obtained in (3.5). The max part is represented as

$$\begin{aligned} \max_{\Sigma_1,\Sigma_2} \ & \frac{1}{2}\log\frac{|\Sigma_X+\Theta||\mathrm{diag}(\Sigma_1,\Sigma_2)|}{|\mathrm{diag}(\Sigma_1,\Sigma_2)+\Theta|\mathrm{diag}(\Gamma_1,\Gamma_2)|} \\ \text{subject to} \ \ & 0 \preceq \mathrm{diag}(\Sigma_1,\Sigma_2) \preceq \Sigma_X \\ & (\mathrm{diag}(\Gamma_1^{-1},\Gamma_2^{-1})-\Theta^{-1})^{-1} \preceq \mathrm{diag}(\Sigma_1,\Sigma_2) \preceq \Sigma_X. \end{aligned} \tag{3.6}$$

As we mentioned before, linearization on the objective function here is necessary, otherwise CVX can not handle it. We introduce the Taylor Series to linearize the

objective function in (3.6) as

$$\begin{aligned} &\max_{\Sigma_1,\Sigma_2} \frac{1}{2}\log|\mathrm{diag}(\Sigma_1,\Sigma_2)| - \frac{1}{2}\mathrm{tr}((\mathrm{diag}(\Sigma_1^{(n)},\Sigma_2^{(n)})+\Theta))^{-1}\cdot\mathrm{diag}(\Sigma_1,\Sigma_2) \\ &\text{subject to } \ 0 \preceq \mathrm{diag}(\Sigma_1,\Sigma_2) \preceq \Sigma_X \\ &\qquad (\mathrm{diag}(\Gamma_1^{-1},\Gamma_2^{-1})-\Theta^{-1})^{-1} \preceq \mathrm{diag}(\Sigma_1,\Sigma_2) \preceq \Sigma_X. \end{aligned} \tag{3.7}$$

After finishing this max part, we have to make some refinement on the matrix Σ_1, Σ_2 obtained at (3.7). Here we introduce one parameter α as the step size, and let the Σ_1 and Σ_2, which are obtained at this present step, be noted as Σ_1^*, Σ_2^*. In addition, we use $\Sigma_1^{(n)}$, $\Sigma_2^{(n)}$ to represent the value of Σ_1, Σ_2 obtained at the last iteration in the same step as (3.7). If it is in the first iteration, $\Sigma_1^{(n)}$ and $\Sigma_2^{(n)}$ will be the initial values of Σ_1, Σ_2 respectively. Therefore, the value of Σ_1 and Σ_2 in the max part we get is

$$\begin{aligned} \Sigma_1 &= (1-\alpha)\Sigma_1^{(n)} + \alpha\cdot\Sigma_1^* \\ \Sigma_2 &= (1-\alpha)\Sigma_2^{(n)} + \alpha\cdot\Sigma_2^*. \end{aligned} \tag{3.8}$$

Now we have already obtained the value of all variables D, Σ_i and Γ_i $(i=1,2)$. And the last step in these procedures is to compute the lower-bound and upper-bound for the minimum sum rate with all the values we got. To get the upper bound here, we have to do the following computation with Γ_1, Γ_2 and Σ_Z as

$$D^* = ((\mathrm{diag}(\Gamma_1,\Gamma_2))^{-1} - \Sigma_Z^{-1})^{-1}. \tag{3.9}$$

Then we put the D^* back to the objective function in (3.5) and get the upper bound as follows

$$\text{upper bound} = \frac{1}{2}\log\frac{|\Sigma_X+\Sigma_Z||\mathrm{diag}(\Sigma_1,\Sigma_2)|}{|D^*+\Sigma_Z||\mathrm{diag}(\Gamma_1,\Gamma_2)|}. \tag{3.10}$$

Similar to the upper-bound, if we put the obtained value of D directly to (3.5), the lower-bound is obtained as

$$\text{lower bound} = \frac{1}{2}\log\frac{|\Sigma_X + \Sigma_Z||\text{diag}(\Sigma_1, \Sigma_2)|}{|D + \Sigma_Z||\text{diag}(\Gamma_1, \Gamma_2)|} \tag{3.11}$$

thus the gap between upper-bound and lower-bound is represented as

$$\frac{1}{2}\log\frac{|\Sigma_X + \Sigma_Z||\text{diag}(\Sigma_1, \Sigma_2)|}{|D^* + \Sigma_Z||\text{diag}(\Gamma_1, \Gamma_2)|} - \frac{1}{2}\log\frac{|\Sigma_X + \Sigma_Z||\text{diag}(\Sigma_1, \Sigma_2)|}{|D + \Sigma_Z||\text{diag}(\Gamma_1, \Gamma_2)|}. \tag{3.12}$$

Now all the process in one iteration is completed. The goal in this algorithm is to reduce the gap between upper-bound and lower-bound and make them converge. After finishing one iteration, we put the obtained value of Σ_1, Σ_2 to the min part in (3.5) again and launch the new iteration. It can be verified from simulations that the gap between upper-bound and lower-bound can be reduced with iterations, and both bounds can get converged when $L = 2$.

3.3 Results of the iterative algorithm

This part will represent some numerical examples to illustrate the iterative algorithm composed by the min part and the max part.

Let $X = (X_1^T, X_2^T)^T$. We will begin with an example with Σ_X and Σ_1, Σ_2. The Σ_X is a randomly generated matrix with dimension 8×8. Here we assume the source vector to be divided by two parts X_1 and X_2, each of them with the dimension 4, X is mean zero and Σ_X is the covariance matrix of X. The dimension of Σ_1 and Σ_2 is the same as the dimension of X_1, X_2 correspondingly.

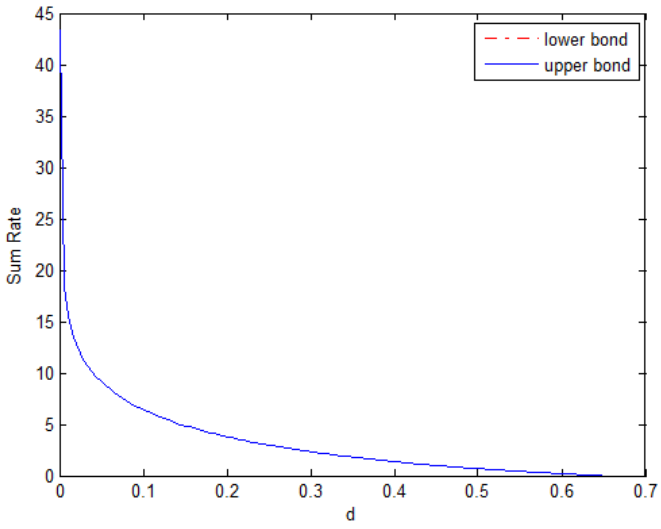

Figure 3.1: The upper-bound and lower-bound when Σ_X is 8×8, X_1 and X_2 with the dimension of 4, $L = 2$.

Fig 3.1 represents the upper-bound and lower-bound in the minimum sum rate. The X-axis indicates the value of distortion d, from 0 to $\mathrm{tr}(\Sigma_X)$. And the Y-axis here is the minimum sum rate corresponding to every value of d. It is obvious that when the dimension of Σ_X is 8, which is not a large scale, the upper-bound and lower-bound can touch each other perfectly.

We want to take the example further. Since in Fig 3.1, Σ_X has a relatively small dimension, now we want to change the dimension to a larger scale, to see if the convergence still exists. Let the dimension of the covariance matrix Σ_X to be 28, which means the vector X is of the dimension 28. We divide it to the vector X_1 and X_2, with dimensions of 13 and 15 respectively.

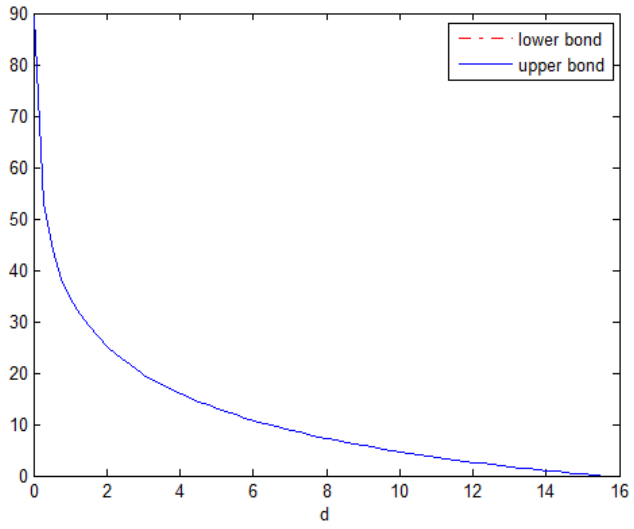

Figure 3.2: The upper-bound and lower-bound when Σ_X is 28×28, X_1 with the dimension of 13, X_2 with dimension of 15, $L = 2$.

The scenario is shown in Fig 3.2. It can be seen that under such circumstance, the lower-bound and upper-bound of the objective function still can touch each other perfectly in all values of distortion d

In the above examples, we assume the vectors X_1 and X_2 are independent, now we want to test if the algorithm still works when X_1 and X_2 are correlated as

$$\begin{aligned} X &= (X_1^T, X_2^T)^T \\ X_2 &= AX_1 + N_2. \end{aligned} \tag{3.13}$$

Where A is a randomly generated matrix and N_2 is a vector, acting as an error term here, with the elements in a much smaller scale than X_1.

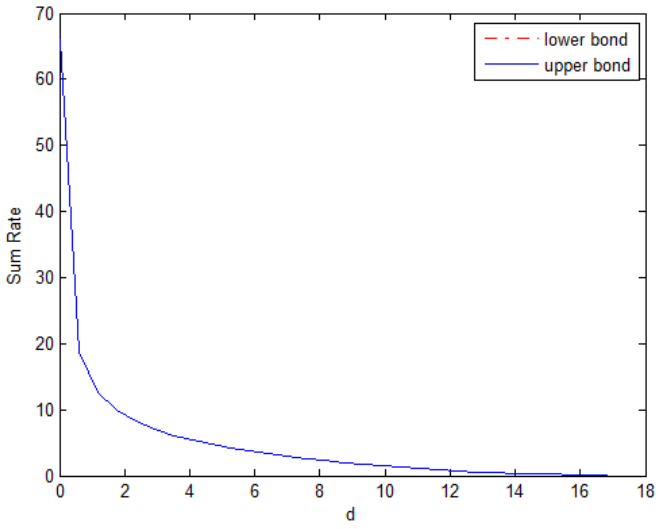

Figure 3.3: The upper-bound and lower-bound when Σ_X is 25×25, X_1 with the dimension of 9, X_2 with the dimension of 16, $L = 2$.

When testing the case with correlation, we set the dimension of Σ_X to be 25, matrix Σ_1, Σ_2 with dimension of 9 and 16. And the vector X_1, X_2 has the relationship in (3.13). Fig 3.3 described the lower-bound and upper bound in this scenario, where both the upper-bound and lower-bound still can get touched, indicating that the iterative algorithm works even when vectors in X have correlation relationship.

However, in the above cases, we define the vectors X_1 and X_2 of the similar numerical value, now we want to test if the element in X_1 has a significant greater value than X_2.

In Fig 3.4, Σ_X is a 19×19 matrix, and X_1 and X_2 have the dimension of 7 and 12. In this scenario, we set the element of X_1 is 15 times greater of those in X_2 in numerical value. As the figure shows, this iterative algorithm is feasible in this situation.

In these examples, including different dimensions of Σ_X and different relationships

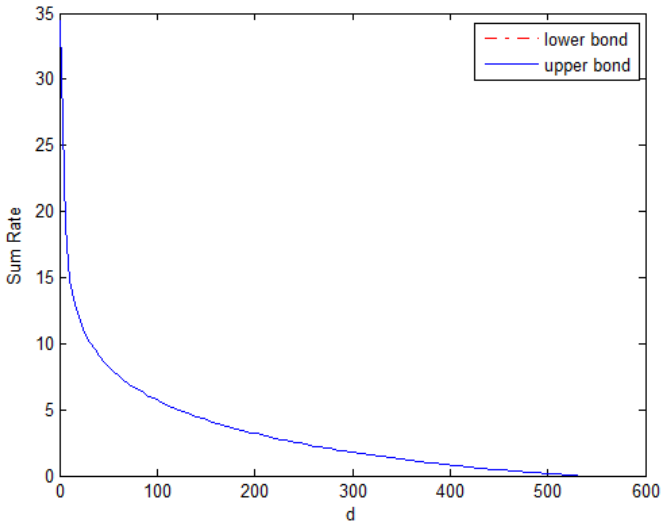

Figure 3.4: The upper-bound and lower-bound when Σ_X is 19×19, X_1 with the dimension of 7, X_2 with the dimension of 12 $L = 2$.

between X_1 and X_2, we can find that the iterative algorithm derived from the saddle point analysis can make the upper-bound and the lower-bound converge when the number of terminals L is 2. And we wonder what the result will be if L is grater than 2.

Let vector $X = (X_1^T, X_2^T, X_3^T)^T$, where X_1, X_2 and X_3 are of the similar numerical value and they are not correlated with each other. The dimensions of these 3 vectors are 4, 7 and 9. Fig 3.5 shows the result of upper-bound and lower-bound in this situation, which indicates that both of them can still converge. In the following example, when the dimension of these three vectors are 12, 3 and 9. Similar to the last case, X_1, X_2 and X_3 are of the similar numerical value and they are not correlated with each other. The lower-bound of the minimum sum rate begins to have the complex numbers when d is less than half of the $\mathrm{tr}(\Sigma_X)$. Besides, the case of $L = 5$ has been tested, and the same situation happens that the lower-bound appears

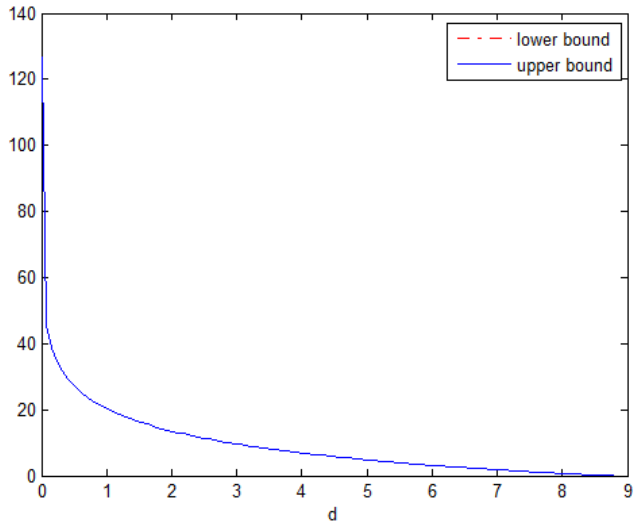

Figure 3.5: The upper-bound and lower-bound when Σ_X is 20×20, X_1, X_2 and X_3 with the dimension of 4, 7, 9, $L = 3$.

to have an imaginary part. It indicates that the toolbox or software we presently utilize may not able to handle the situation perfectly when the number of terminals is greater than 2, thus the error term exists. Or this algorithm may not suit the situation when L is greater than 2 in a general way.

Chapter 4

The Algorithm with Wyner's Common Information

4.1 The connection with Wyner's common information

In Chapter 3, we have presented an iterative algorithm composed of two parts. In each iteration, the value of all variables will be updated and pass to the next iteration for computation, until the lower-bound and upper-bound converge.

It is clear that the value of matrix Σ_i (in the following part, we assume $L = 2$, so i=1,2) is of significant importance. Numerical results suggest that if a particular choice of (Σ_1, Σ_2) produces a tight bound on the rate-distortion function (3.5) in a given d, then the same pair (Σ_1, Σ_2) is also able to make the upper-bound and lower-bound touch for $d' < d$ in certain scenarios. It simulates the interest to know what kind of (Σ_1, Σ_2) is optimal when $d \approx \mathrm{tr}(\Sigma_X)$.

To realize this idea, we can revisit the following optimization problem, which has been mentioned in Chapter 3, as the max part of the iterative problem

$$\begin{aligned} &\max_{\Sigma_1,\Sigma_2} \frac{1}{2}\log\frac{|\Sigma_X+\Theta||\mathrm{diag}(\Sigma_1,\Sigma_2)|}{|\mathrm{diag}(\Sigma_1,\Sigma_2)+\Theta|\mathrm{diag}(\Gamma_1,\Gamma_2)|} \\ &\text{subject to}\ \ 0\preceq \mathrm{diag}(\Sigma_1,\Sigma_2)\preceq\Sigma_X \\ &\qquad\qquad (\mathrm{diag}(\Gamma_1^{-1},\Gamma_2^{-1})-\Theta^{-1})^{-1}\preceq\mathrm{diag}(\Sigma_1,\Sigma_2)\preceq\Sigma_X. \end{aligned} \tag{4.1}$$

Note that when $d\approx\mathrm{tr}(\Sigma_X)$, the diagonal entries of Θ always go to infinity. Since

$$\Theta=(D^{-1}-\Sigma_X^{-1})^{-1}.$$

As a consequence, the result of $|\mathrm{diag}(\Sigma_1,\Sigma_2)+\Theta|$ essentially dose not depend on (Σ_1, Σ_2). However, if we further ignore the constraint for the max part $(\mathrm{diag}(\Gamma_1^{-1},\Gamma_2^{-1})-\Theta^{-1})^{-1}\preceq\mathrm{diag}(\Sigma_1,\Sigma_2)$, then we can rewrite the above optimization problem as

$$\begin{aligned} &\max_{\Sigma_1,\Sigma_2} \frac{1}{2}\log|\mathrm{diag}(\Sigma_1,\Sigma_2)| \\ &\text{subject to}\ \ 0\preceq\mathrm{diag}(\Sigma_1,\Sigma_2)\preceq\Sigma_X. \end{aligned} \tag{4.2}$$

From the form we derived in (4.1), it is quite interesting to note that this is exactly the optimization problem for determining Wyner's common information [17] in the Gaussian case. Specially, let $X=(X_1^T,X_2^T)^T$, then the optimization problem (4.1) will be equivalent to

$$\begin{aligned} &\min_{p_{W|X_1,X_2}} I(X_1,X_2;W) \\ &\text{subject to}\ \ X_1\leftrightarrow W\leftrightarrow X_2 \quad \text{from a Markov chain.} \end{aligned} \tag{4.3}$$

Let the pair of matrix (Σ_1,Σ_2) denote the optimal solution to the optimization problem in (4.1). Under such circumstance, it can be verified that the equation $\text{diag}(\Sigma_1^{-1}, \Sigma_2^{-1}) - \Sigma_X^{-1}$ must be rank deficient.

Therefore, $\Sigma_Z \triangleq (\text{diag}(\Sigma_1^{-1}, \Sigma_2^{-1}) - \Sigma_X^{-1})$ will not be well defined. However, this issue can be easily handled through a suitable projection to the non-degenerate subspace. Now let $\alpha_1, ..., \alpha_p$ denote the positive eigenvalues of the result $\text{diag}(\Sigma_1^{-1}, \Sigma_2^{-1}) - \Sigma_X^{-1}$, and let $\pi_1, ..., \pi_p$ be the corresponding eigenvectors.

We can rewrite the objective function

$$\frac{1}{2}\log\frac{|\Sigma_X + \Sigma_Z||\text{diag}(\Sigma_1, \Sigma_2)|}{|D + \Sigma_Z||\text{diag}(\Gamma_1, \Gamma_2)|}$$

as

$$\frac{1}{2}\log\frac{|\Pi^T\Sigma_X\Pi + \Lambda||\text{diag}(\Sigma_1, \Sigma_2)|}{|\Pi^T D\Pi + \Lambda||\text{diag}(\Gamma_1, \Gamma_2)|}. \tag{4.4}$$

The constraint can be written as

$$0 \preceq \begin{pmatrix} \Pi^T D\Pi + \Lambda & \Pi^T D \\ D\Pi & D - \text{diag}(\Gamma_1, \Gamma_2) \end{pmatrix},$$

where we define

$$\Lambda \triangleq \text{diag}(\frac{1}{\alpha_1}, ..., \frac{1}{\alpha_p})$$

$$\Pi \triangleq (\pi_1, ..., \pi_p).$$

4.2 The procedure of the algorithm

The main difference between the algorithm with Wyner's common information and the iterative algorithm mentioned in Chapter 3 lies in the initialization of the matrix pair (Σ_1, Σ_2).

We start with an optimization problem over Σ_1 and Σ_2. Instead of generating a random matrix as Σ_1 and Σ_2, we try to initialize these two matrix through an optimization problem as follows

$$\begin{aligned} &\max_{\Sigma_1,\Sigma_2} \frac{1}{2}\log|\mathrm{diag}(\Sigma_1,\Sigma_2)| \\ &\text{subject to } \ 0 \preceq \mathrm{diag}(\Sigma_1,\Sigma_2) \preceq \Sigma_X. \end{aligned} \tag{4.5}$$

After obtaining the value of initial Σ_1 and Σ_2, we will start the main part of this algorithm, which looks quite similar to the iterative algorithm in Chapter 3, except some modifications when it concerns Σ_Z.

We will divide the algorithm by the min part and the max part, and solve the problem in an iterative way. In the min part, we will fix Σ_1 and Σ_2 to the correct values and try to obtain the value of D, Γ_1 and Γ_2. The min part optimization is

written as follows,

$$\begin{aligned}
&\min_{D,\Gamma_1,\Gamma_2} \frac{1}{2}\log\frac{|\Pi^T\Sigma_X\Pi+\Lambda||\mathrm{diag}(\Sigma_1,\Sigma_2)|}{|\Pi^T D\Pi+\Lambda||\mathrm{diag}(\Gamma_1,\Gamma_2)|}\\
&\text{subject to}\ \ 0\preceq D\preceq\Sigma_X,\\
&\qquad\qquad \mathrm{tr}(D)\le d,\\
&\qquad\qquad 0\preceq \mathrm{diag}(\Gamma_1,\Gamma_2),\\
&\qquad\qquad 0\preceq \begin{pmatrix}\Pi^T D\Pi+\Lambda & \Pi^T D\\ D\Pi & D-\mathrm{diag}(\Gamma_1,\Gamma_2)\end{pmatrix}
\end{aligned} \tag{4.6}$$

where $\alpha_1,...,\alpha_p$ denote the positive eigenvalues of the result $\mathrm{diag}(\Sigma_1^{-1},\Sigma_2^{-1})-\Sigma_X^{-1}$, and $\pi_1,...,\pi_p$ denote the corresponding eigenvectors. We define $\Lambda\triangleq\mathrm{diag}(\frac{1}{\alpha_1},...,\frac{1}{\alpha_p})$, $\Pi\triangleq(\pi_1,...,\pi_p)$.

Then we start the max part, obtaining the maximum value of the objective function from the variables Σ_1 and Σ_2. Here we treat the variables D, Γ_1 and Γ_2 as constants. The max part is represented as

$$\begin{aligned}
&\max_{\Sigma_1,\Sigma_2} \frac{1}{2}\log\frac{|\Sigma_X+\Theta||\mathrm{diag}(\Sigma_1,\Sigma_2)|}{|\mathrm{diag}(\Sigma_1,\Sigma_2)+\Theta||\mathrm{diag}(\Gamma_1,\Gamma_2)|}\\
&\text{subject to}\ \ 0\preceq\mathrm{diag}(\Sigma_1,\Sigma_2)\preceq\Sigma_X\\
&\qquad\qquad (\mathrm{diag}(\Gamma_1^{-1},\Gamma_2^{-1})-\Theta^{-1})^{-1}\preceq\mathrm{diag}(\Sigma_1,\Sigma_2)\preceq\Sigma_X
\end{aligned} \tag{4.7}$$

see $\Theta=(D^{-1}-\Sigma_X^{-1})^{-1}$. And the same linearization function as appeared in Chapter 3, (3.7), should be introduced in this max part.

Now after obtaining the value of all variables D, Σ_i and Γ_i $(i=1,2)$, the last step in these procedures is to compute the lower-bound and upper-bound for the minimum sum rate with all the values we got. To get the upper bound here, we have to do the

following computation with Γ_1, Γ_2 and Π, Λ as

$$D^* = ((\text{diag}(\Gamma_1, \Gamma_2))^{-1} - \Pi\Lambda^{-1}\Pi^T. \tag{4.8}$$

Then we put the D^* to the objective function in (4.6) and get the upper bound

$$\text{upper bound} = \frac{1}{2}\log\frac{|\Pi^T\Sigma_X\Pi + \Lambda||\text{diag}(\Sigma_1, \Sigma_2)|}{|\Pi^T D^*\Pi + \Lambda||\text{diag}(\Gamma_1, \Gamma_2)|}. \tag{4.9}$$

Similar to the upper-bound, if we put the obtained value of D directly to (4.6), the lower-bound is obtained as

$$\text{lower bound} = \frac{1}{2}\log\frac{|\Pi^T\Sigma_X\Pi + \Lambda||\text{diag}(\Sigma_1, \Sigma_2)|}{|\Pi^T D\Pi + \Lambda||\text{diag}(\Gamma_1, \Gamma_2)|} \tag{4.10}$$

thus the gap between upper-bound and lower-bound is represented as

$$\frac{1}{2}\log\frac{|\Pi^T\Sigma_X\Pi + \Lambda||\text{diag}(\Sigma_1, \Sigma_2)|}{|\Pi^T D^*\Pi + \Lambda||\text{diag}(\Gamma_1, \Gamma_2)|} - \frac{1}{2}\log\frac{|\Pi^T\Sigma_X\Pi + \Lambda||\text{diag}(\Sigma_1, \Sigma_2)|}{|\Pi^T D\Pi + \Lambda||\text{diag}(\Gamma_1, \Gamma_2)|}. \tag{4.11}$$

The above is all the procedures in one iteration. Similar to the iterative algorithm in Chapter 3, the goal in this algorithm with Wyner's common information is to reduce the gap between upper-bound and lower-bound and make them converge. With the initialization of Σ_1 and Σ_2 at first, the operation efficiency of this algorithm can be improved.

Therefore, after finishing one iteration, we put the obtained value Σ_1, Σ_2 to the min part in (4.6) again and start the new iteration.

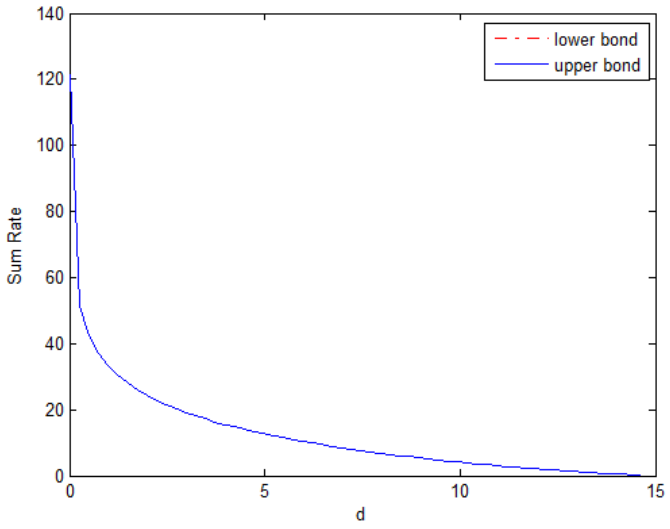

Figure 4.1: The upper-bound and lower-bound when Σ_X is 26×26, $L = 2$. X_1 with the dimension of 10, X_2 with the dimension of 16.

4.3 Results of this algorithm

This part will represent some numerical examples for the algorithm with Wyner's common information.

Let $X = (X_1^T, X_2^T)^T$. We will begin with an example with Σ_X and Σ_1, Σ_2. The Σ_X is a randomly generated matrix with dimension 26×26. Here we assume the source vector to be divided by two parts X_1 and X_2, each with the the dimension of 10 and 16, X is mean zero and Σ_X is the covariance matrix of X. The dimension of Σ_1 and Σ_2 is the same as the dimension of X_1, X_2 correspondingly.

Fig 4.1 represents the upper-bound and lower-bound in the minimum sum rate. The X-axis indicates the value of distortion d, from 0 to $tr(\Sigma_X)$. The Y-axis here is the minimum sum rate corresponding to every value of d. It can be observed that the bounds have converged. We want to take the example further.

We now want to test if the algorithm still works when X_1 and X_2 are correlated

$$\begin{aligned} X &= (X_1^T, X_2^T)^T \\ X_2 &= AX_1 + N_2. \end{aligned} \tag{4.12}$$

Where A is a randomly generated matrix and N_2 is a vector, acting as an error term here, with the elements in a much smaller scale than X_1.

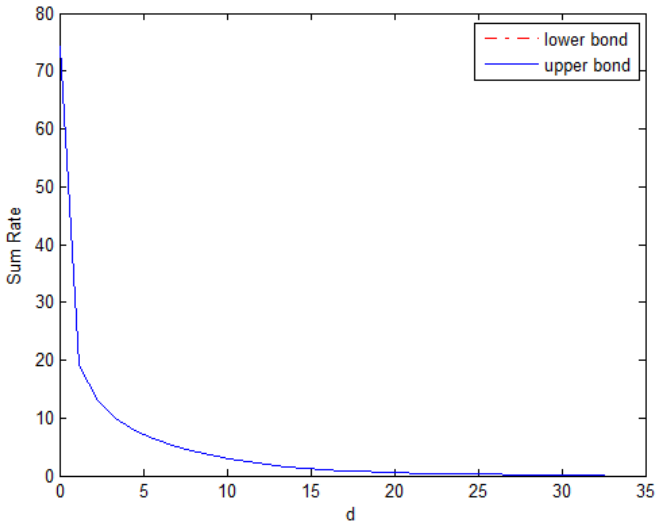

Figure 4.2: The upper-bound and lower-bound when Σ_X is 25×25, $L = 2.X_1$ with the dimension of 9, X_2 with the dimension of 16.

When testing the case with correlation, we set the dimension of Σ_X to be 25, matrix Σ_1, Σ_2 with dimension of 9 and 16. And the vectors X_1, X_2 has the relationship in (4.14). Fig 4.2 described the lower-bound and upper-bound in this scenario, where both the upper-bound and lower-bound still can get touched, indicating that this new iterative algorithm with Wyner's common information works even when vector X has correlation.

However, in the above cases, we define the vectors X_1 and X_2 are of the similar numerical value and dimension, now we want to test if the element in X_1 has a significant larger value than X_2.

In Fig 4.3, Σ_X is a 24×24 matrix, and X_1 and X_2 has the dimension of 3 and 21. In this scenario, we set the element of X_1 is 20 times larger of those in X_2 in numerical value. As the figure shows, this algorithm is feasible in this situation.

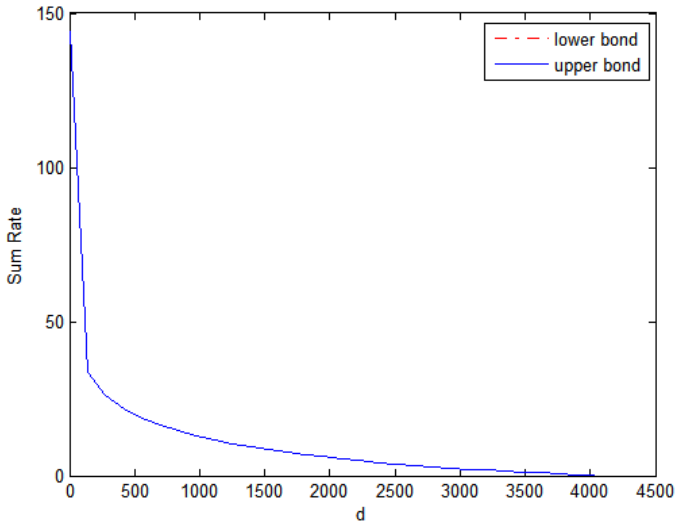

Figure 4.3: The upper-bound and lower-bound when Σ_X is 24×24, $L = 2.X_1$ with the dimension of 3, X_2 with the dimension of 21.

After testing the examples, including different dimension of Σ_X and different relationships between X_1 and X_2, we can draw the conclusion that the iterative algorithm with Wyner's common information works when the number of terminals L is 2. And we wonder how this algorithm will perform if L is grater than 2.

Let vector $X = (X_1^T, X_2^T, X_3^T)^T$, where X_1, X_2 and X_3 are of the similar numerical value and they are not correlated with each other. The dimensions of these 3 vectors are 3, 8 and 6. Fig 4.5 shows the result of upper-bound and lower-bound in this situation, which indicates that the bounds converge.

In the following example, the dimension of these three vectors are 3, 8 and 6, the same as the last example. Here X_1, X_2 and X_3 are of the similar numerical value but they are correlated with each other. When d is in a certain range, the gap between upper-bound and lower-bound can not be ignored even increasing the number of iterations, the gap can be seen in Fig 4.5. It seems that there are cases

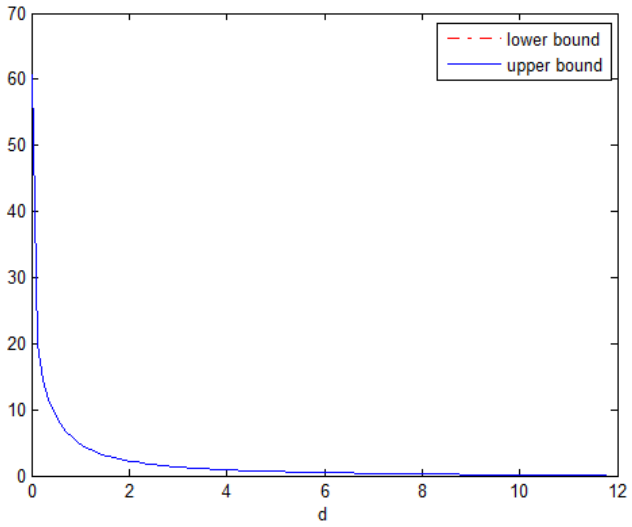

Figure 4.4: The upper-bound and lower-bound when Σ_X is 17×17, $L = 3$. X_1 with the dimension of 3, X_2 with the dimension of 8, X_3 with with the dimension of 6.

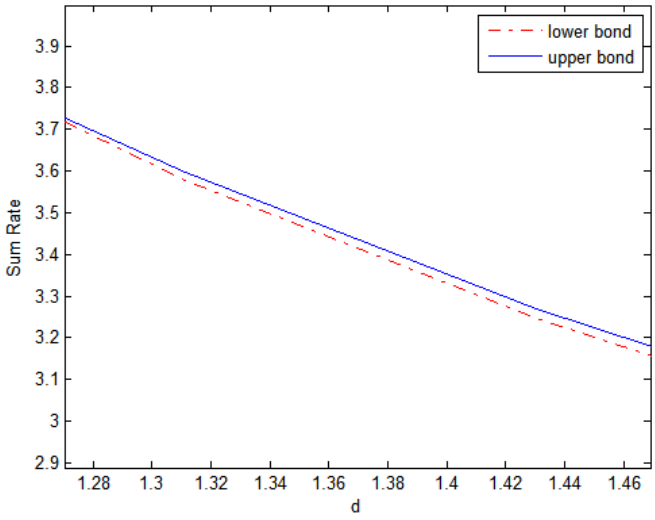

Figure 4.5: The upper-bound and lower-bound when Σ_X is 17×17, $L = 3$. X_1 with the dimension of 3, X_2 with the dimension of 8, X_3 with with the dimension of 6, correlated.

with three terminals where this algorithm with Wyner's common information is not able to reduce the gap completely.

Chapter 5

The Algorithm with Initializations

5.1 Methods to initialize the variable

In this chapter, we will derive three methods to initialize one key variable Σ_Z, which has appeared in Chapters 3 and 4. In the previous chapters, we define Σ_Z as follows

$$\Sigma_Z = ((\text{diag}(\Sigma_1, \cdots, \Sigma_L))^{-1} - \Sigma_X^{-1})^{-1} \tag{5.1}$$

then we launch the algorithms iteratively until the upper-bound and lower-bound converge. However, we find that if we initialize the value of Σ_Z at first and derive $\Sigma_i, (i = 1, \cdots, L)$ from the obtained Σ_Z, sometimes it is also feasible to make the upper-bound and lower-bound touched. Under such circumstance, we can avoid the iterative procedures. Instead, we put the Σ_i derived from Σ_Z directly to the optimization problem for computing the minimum sum rate. In the following part, we assume $L = 2$.

We can derive Σ_Z directly from the covariance matrix Σ_X as follows

$$\Sigma_Z = \begin{pmatrix} \Sigma_{X_{11}} & -\Sigma_{X_{12}} \\ -\Sigma_{X_{21}} & \Sigma_{X_{22}} \end{pmatrix} \tag{5.2}$$

where $\Sigma_{X_{11}}$ denotes to the upper left matrix in D with dimension $i \times i$, where i denotes the dimension of vector X_1. Let j denotes the dimension of X_2. Thus $\Sigma_{X_{12}}$ denotes the part of Σ_X from column $i+1$ to column $i+j$ and row 1 to row i .The rest is represented in the same manner.

After obtaining the value of Σ_Z, we can derive (Σ_1,Σ_2) by the equation

$$\text{diag}(\Sigma_1, \Sigma_2) = (\Sigma_Z^{-1} + \Sigma_X^{-1})^{-1} \tag{5.3}$$

then we start the optimization problem for computing the minimum sum rate, by putting the derived Σ_1 and Σ_2 to it directly

$$\begin{aligned} \min_{D,\Gamma_1,\Gamma_2} \ & \frac{1}{2} \log \frac{|\Sigma_X + \Sigma_Z||\text{diag}(\Sigma_1, \Sigma_2)|}{|D + \Sigma_Z||\text{diag}(\Gamma_1, \Gamma_2)|} \\ \text{subject to} \ \ & 0 \preceq D \preceq \Sigma_X, \\ & \text{tr}(D) \leq d, \\ & 0 \preceq \text{diag}(\Gamma_1, \Gamma_2), \\ & \text{diag}(\Gamma_1, \Gamma_2) \preceq (D^{-1} + \Sigma_Z^{-1})^{-1}. \end{aligned} \tag{5.4}$$

The approach to compute the upper-bound and lower-bound is the same as in Chapter 3, the gap can be represented as

$$\frac{1}{2} \log \frac{|\Sigma_X + \Sigma_Z||\text{diag}(\Sigma_1, \Sigma_2)|}{|D^* + \Sigma_Z||\text{diag}(\Gamma_1, \Gamma_2)|} - \frac{1}{2} \log \frac{|\Sigma_X + \Sigma_Z||\text{diag}(\Sigma_1, \Sigma_2)|}{|D + \Sigma_Z||\text{diag}(\Gamma_1, \Gamma_2)|} \tag{5.5}$$

where

$$D^* = ((\text{diag}(\Gamma_1, \Gamma_2))^{-1} - \Sigma_Z^{-1})^{-1}. \tag{5.6}$$

Now we do not need to complete another max part as in Chapters 3 and 4. Instead, we compute the gap from one optimization problem(5.4) directly.

Apart from deriving Σ_Z from Σ_X, we also can derive Σ_Z from an optimal D as

$$\Sigma_Z = \begin{pmatrix} D_{11} & -D_{12} \\ D_{21} & D_{22} \end{pmatrix} \tag{5.7}$$

where D_{11} denotes to the upper left part in D with dimension $i \times i$. Let j denotes the dimension of X_2. Thus D_{12} denotes to the part in D from column $i+1$ to column $i+j$ and row 1 to row i .The rest is represented in the same manner. The following procedures are the same as mentioned before, from (5.3) to (5.5).

What's more, we also can derive Σ_Z from the computed D^* in the similar way as

$$\Sigma_Z = \begin{pmatrix} D_{11}^* & -D_{12}^* \\ D_{21}^* & D_{22}^* \end{pmatrix} \tag{5.8}$$

and proceed the algorithm as the last two mentioned approaches(5.2) and (5.7).

5.2 Results

In this section we will represent some examples to show the results of those three algorithms.

We will test these three algorithms with the same matrix Σ_X and the same corresponding dimensions in Σ_1, Σ_2.

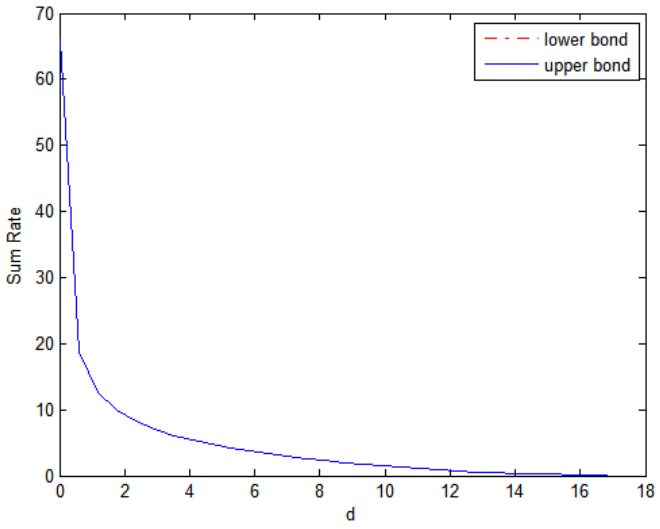

Figure 5.1: The upper-bound and lower-bound when Σ_X is 25×25, $L = 2$, derive Σ_Z from Σ_X. X_1 with the dimension of 9, X_2 with the dimension of 16.

We set Σ_X with dimension 25×25, and vector X_1 with the dimension of 9, vector X_2 with the dimension of 16. X_1 and X_2 are correlated, which has the relationship in (4.14). Fig 5.1 represents the upper-bound and lower-bound when computing the Σ_Z from Σ_X, which shows that the two bounds get converge. The figure for the other two algorithms, which compute Σ_Z from D and D^*, have nearly the same figure as Fig 5.1.

In the Fig 5.1, we take the value of distortion d as the X-axis, from 0 to $\mathrm{tr}(\Sigma_X)$. Here we pick 30 points uniformly to draw the figure. However, there exists a part in d, where all these three algorithms produce the non-negligible gap between lower-bound and upper-bound. In order to show the gap numerically and make the comparison more directly, I will represent the normalized gap in following table 5.1. The gap will

be normalized as

$$\text{normalized gap} = \frac{\text{upper-bound} - \text{lower-bound}}{\text{upper-bound}}. \tag{5.9}$$

Table 5.1: The Gap between lower-bond and upper-bound in three algorithms, when X_1 with dimension 9 and X_2 with dimension 16

d	2.9165	3.4975	4.079	4.660	5.241	5.822	6.404	6.985
Gap 1	9.21×10^{-8}	2.78×10^{-5}	2.75×10^{-4}	2.64×10^{-4}	2.36×10^{-4}	2.08×10^{-4}	1.76×10^{-4}	1.28×10^{-4}
Gap 2	5.79×10^{-4}	4.07×10^{-4}	3.56×10^{-4}	3.15×10^{-4}	2.78×10^{-4}	2.44×10^{-4}	2.08×10^{-4}	1.83×10^{-4}
Gap 3	5.79×10^{-4}	4.07×10^{-4}	3.56×10^{-4}	3.15×10^{-4}	2.78×10^{-4}	2.44×10^{-4}	2.08×10^{-4}	1.83×10^{-4}

In the above Table 5.1, gap 1 denotes the gap when computing Σ_Z from Σ_X in (5.2), and gap 2 and 3 denote to the scenarios where Σ_Z is derived from D (5.7) and D^*(5.8). It can be seen from Table 5.1 that all those three algorithms produce a relatively obvious gap in the same range of d, with a normalized gap on the scale of 10^{-4}. The first algorithm appears to perform better than the other two, but only in a slight degree. Algorithm 2 and 3 seem to have the same performance in this scenario, with nuanced difference as follows, where ΔGap denotes the difference between Gap 2 and Gap 3.

Table 5.2: The Gap difference in Gap 2 and Gap 3, when X_1 with dimension 9 and X_2 with dimension 16

d	2.9165	3.4975	4.079	4.660	5.241	5.822	6.404	6.985
ΔGap	-1.46×10^{-7}	-3.063×10^{-7}	-3.89×10^{-7}	7.50×10^{-7}	-3.17×10^{-7}	2.41×10^{-7}	2.84×10^{-7}	3.16×10^{-7}

We can test another example when Σ_X with dimension 25×25, the dimension of X_1 is 12, and the dimension of X_2 is 13. We wonder what the performance of those three algorithms will be when X_1 and X_2 share the similar dimensions. Vector X_1

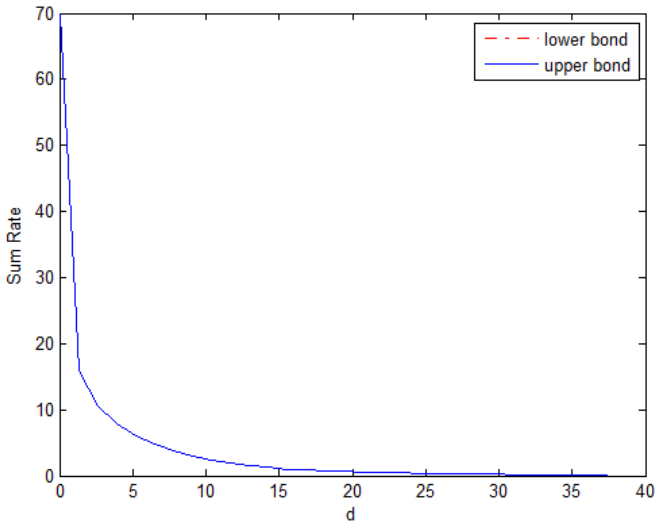

Figure 5.2: The upper-bound and lower-bound when Σ_X is 25×25, $L = 2$, derive Σ_Z from Σ_X. X_1 with the dimension of 12, X_2 with the dimension of 13.

and X_2 are correlated. Like Fig 5.1, we will represent the figure of upper-bound and lower-bound in algorithm one when the value of d ranges from 0 to $\mathrm{tr}(\Sigma_X)$ as follows.

However, there also exists a certain range in d where the gap can not be reduced perfectly. We will represent the normalized gap between upper-bound and lower-bound in that part to show the results and compare the performance between these three algorithms.

Table 5.3: The Gap between lower-bond and upper-bound in three algorithms, when X_1 with dimension 12 and X_2 with dimension 13

d	3.885	5.176	6.468	7.759	9.051	10.343	11.634	12.926
Gap 1	2.17×10^{-7}	4.12×10^{-4}	1.37×10^{-3}	1.16×10^{-3}	1.08×10^{-3}	3.28×10^{-3}	3.39×10^{-3}	2.18×10^{-3}
Gap 2	2.22×10^{-3}	6.55×10^{-3}	7.04×10^{-3}	6.72×10^{-3}	1.16×10^{-3}	3.44×10^{-3}	5.50×10^{-4}	5.28×10^{-4}
Gap 3	2.22×10^{-3}	6.55×10^{-3}	7.04×10^{-3}	6.72×10^{-3}	1.17×10^{-3}	3.45×10^{-3}	5.49×10^{-4}	5.28×10^{-4}

It seems that in this scenario, when X_1 and X_2 share almost the same dimension

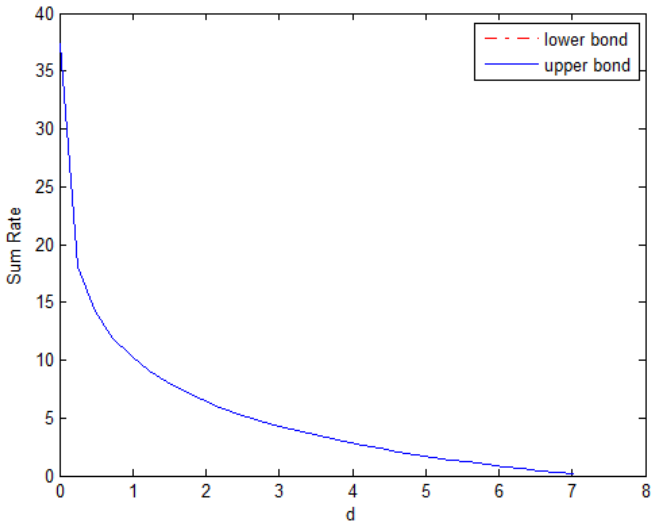

Figure 5.3: The upper-bound and lower-bound when Σ_X is 12×12, $L = 2$, derive Σ_Z from Σ_X. X_1 with the dimension of 5, X_2 with the dimension of 7.

and are correlated, those three algorithms represent nearly the same performance and those gaps are not negligible. It can be seen that the first algorithm performs better than the other two in this certain range, when gap 2 and 3 are quite obvious. Algorithms 2 and 3 still share the similar performance. However, apart from this range of d in the scenario, algorithm 2 and 3 performs better than algorithm one. So it is hard for us to judge which one is better, it depends on the scenario we meet.

When the dimensions of X_1 and X_2 are of the relatively small value and those two vectors are not correlated with each other, all those three algorithms can make the upper-bound and lower-bound touched closely. We can find that in the following scenario, when the dimensions of X_1, X_2 are 5 and 7. Here both vectors are not correlated with each other.

Fig 5.3 represents the upper-bound and lower-bound in this scenario, when generating Σ_Z from the covariance matrix Σ_X. In order to show the results in a more

detailed way, we will list a part of the normalized gaps as follows.

Table 5.4: The Gap between lower-bond and upper-bound in three algorithms, when X_1 with dimension 5 and X_2 with dimension 7

d	1.707	1.949	2.191	2.434	2.676	2.918	3.161	3.403
Gap 1	1.94×10^{-10}	1.31×10^{-9}	1.07×10^{-10}	4.73×10^{-10}	5.19×10^{-9}	4.61×10^{-9}	7.27×10^{-9}	1.38×10^{-9}
Gap 2	7.42×10^{-10}	2.39×10^{-9}	6.16×10^{-10}	2.42×10^{-10}	6.46×10^{-9}	6.61×10^{-9}	3.60×10^{-9}	4.20×10^{-9}
Gap 3	1.18×10^{-9}	1.59×10^{-10}	7.39×10^{-10}	2.18×10^{-10}	4.12×10^{-9}	7.79×10^{-9}	7.02×10^{-9}	1.07×10^{-9}

In the above Table 5.4, all the three algorithms produce satisfying results. The normalized gaps can be reduced to a magnitude of 10^{-9} or even less in this scenario, which indicates that all the three algorithms perform well under such circumstance.

Chapter 6

Conclusion

In this thesis, we aim at achieving the minimum sum rate in the multi-terminal Gaussian case. To begin with, we derive the equation for minimum sum rate from the rate-distortion function. Then we make conversions of the original optimal problem, transforming it to a convex one and raising an iterative algorithm to converge the upper-bound and lower-bound of the objective function. When those two bounds coincide, it can be verified that we obtain the optimal solution to the minimum sum rate.

We come up with three major kinds of algorithms. The first one is the iterative algorithm derived from the objective function directly. The iterative algorithm is composed of two parts, which aim at obtaining the maximum value and the minimum value of the transformed objective problem. That is to say, the essential of this algorithm is the saddle point analysis.

Based on the first algorithm, we raise another algorithm with Wyner's common information. In this algorithm, we solve an optimal problem at first and obtain the initial value of two fundamental variables. Then we continue the iterative procedures

and get the optimal solution.

We also put forward algorithms without iterations, which generate the variables from known variables and put them back to the objective problem. Under such circumstance, the upper-bound and lower-bound can be computed directly.

All those algorithms can make the lower-bound and upper-bound touch in certain cases, for the gap can be reduced to a relatively small value comparing to the value of upper-bound and lower-bound. The performance of the algorithms will differ in the scenarios they meet, with different dimensions and relationships between the vectors. When the number of encoders L is equal to 2, all those algorithms are applicable. But when L is greater than 2, these series of algorithms may not work well. There could appear numerical errors or the upper-bound and lower-bound may not converge.

Therefore, there is some future work for the cases with three or more terminals. In the aspect of implementation, we can try to solve the optimization problems with another kind of software or toolbox, which are able to handle the problem as well as avoid the numerical errors. Thus the complex number in the lower-bound can be removed. In the algorithm part, We can also try to improve the algorithm by modifying the objective functions and the constraints, to make it more applicable and precise when L is greater than 2.

Appendix A

Derivation of Equation 2.9

We can define the matrix $\Sigma = \text{diag}(\Sigma_1, \cdots, \Sigma_L)$ and $\Gamma = \text{diag}(\Gamma_1, \cdots, \Gamma_L)$ for convenience. Then, we can apply the matrix inversion lemma [18] in the expression for D in (2.4), which will derive the expression for D^{-1} as follows,

$$\begin{aligned} D^{-1} &= \Sigma_X^{-1} C^{-1} (C\Sigma_X C^T + \Sigma_Q)(C\Sigma_X C^T + \Sigma_Q - C\Sigma_X \Sigma_X^{-1} \Sigma_X C^T)^{-1} C\Sigma_X \Sigma_X^{-1} \\ &= \Sigma_X^{-1} C^{-1} (C\Sigma_X C^T + \Sigma_Q)\Sigma_Q^{-1} C \\ &= C\Sigma_Q^{-1} C^T + \Sigma_X^{-1}. \end{aligned} \tag{A.1}$$

From the expression for Σ_Z in (2.7), it can be derived that

$$\begin{aligned} D^{-1} + \Sigma_Z^{-1} &= C\Sigma_Q^{-1} C^T + \Sigma_X^{-1} + \Sigma^{-1} - \Sigma_X^{-1} \\ &= C\Sigma_Q^{-1} C^T + \Sigma^{-1}. \end{aligned} \tag{A.2}$$

The definition of $\Gamma_i, i = 1, ..., L$ is represented in (2.8), Thus the expression of matrix Γ can be proved to be

$$\Gamma = \Sigma - \Sigma C^T (C\Sigma C^T + \Sigma_Q)^{-1} C\Sigma. \tag{A.3}$$

By applying the matrix inversion lemma again, we can get that

$$\Gamma^{-1} = C\Sigma_Q^{-1} C^T + \Sigma^{-1}. \tag{A.4}$$

Therefore, (A..4) shares the same result in (A..2), which proves that

$$\Gamma^{-1} = D^{-1} + \Sigma_Z^{-1} \tag{A.5}$$

is the same as

$$\text{diag}(\Gamma_1, \cdots, \Gamma_L) = (D^{-1} + \Sigma_Z^{-1})^{-1}. \tag{A.6}$$

Appendix B

Derivation of Equation 2.10

$$\begin{aligned}
\frac{1}{2}\log\frac{|C\Sigma_X C^T+\Sigma_Q|}{|\Sigma_Q|} &= \frac{1}{2}\log\frac{|C\Sigma_X C^T+\Sigma_Q|\cdot|\Sigma_Q^{-1}|}{|\Sigma_Q|\cdot|\Sigma_Q^{-1}|}\\
&= \frac{1}{2}\log\frac{|C\Sigma_X\Sigma_Q^{-1}C^T+I|}{|\Sigma_Q|\cdot|\Sigma_Q|^{-1}}\\
&= \frac{1}{2}\log|C\Sigma_X\Sigma_Q^{-1}C^T+I|\\
&= \frac{1}{2}\log(|C\Sigma_Q^{-1}C^T+\Sigma_X^{-1}|\cdot|\Sigma_X|)\\
&= \frac{1}{2}\log\left(\frac{|C\Sigma_Q^{-1}C^T+\Sigma_X^{-1}|}{|\Sigma_X^{-1}|}\right).
\end{aligned} \tag{B.7}$$

We can define the matrix $\Sigma=\text{diag}(\Sigma_1,\cdots,\Sigma_L)$ and $\Gamma=\text{diag}(\Gamma_1,\cdots,\Gamma_L)$. From the definition of Σ_Z in (2.7), we can derive the expression for Σ_X^{-1} as

$$\begin{aligned}
\Sigma_X^{-1} &= \text{diag}(\Sigma_1,\cdots,\Sigma_L)^{-1}-\Sigma_Z^{-1}\\
&= \Sigma^{-1}-\Sigma_Z^{-1}.
\end{aligned} \tag{B.8}$$

As for the numerator in (B..7), we can apply the matrix inversion lemma directly to

it as

$$(C\Sigma_Q^{-1}C^T + \Sigma_X^{-1})^{-1} = \Sigma_X - \Sigma_X C^T (C\Sigma_X C^T + \Sigma_Q)^{-1} C\Sigma_X. \tag{B.9}$$

It is obvious that the conversion of $C\Sigma_Q^{-1}C^T + \Sigma_X^{-1}$ in (B..9) is exactly the definition of matrix D in (2.4). What's more, it has been proved that $\Gamma = (D^{-1} + \Sigma_Z^{-1})^{-1}$. Therefore, (B..7) can be represented as

$$\begin{aligned}
\frac{1}{2}\log\frac{|C\Sigma_X C^T + \Sigma_Q|}{|\Sigma_Q|} &= \frac{1}{2}\log\left(\frac{|D^{-1}|}{|\Sigma^{-1} - \Sigma_Z^{-1}|}\right) \\
&= \frac{1}{2}\log\left(\frac{|\Gamma^{-1} - \Sigma_Z^{-1}|}{|\Sigma^{-1} - \Sigma_Z^{-1}|}\right) \\
&= \frac{1}{2}\log\left(\frac{|I - \Sigma_Z^{-1}\Gamma| \cdot |\Gamma^{-1}|}{|I - \Sigma_Z^{-1}\Sigma| \cdot |\Sigma^{-1}|}\right) \\
&= \frac{1}{2}\log\left(\frac{|\Sigma_Z^{-1} - \Sigma_Z^{-1}\Gamma\Sigma_Z^{-1}| \cdot |\Gamma^{-1}|}{|\Sigma_Z^{-1} - \Sigma_Z^{-1}\Sigma\Sigma_Z^{-1}| \cdot |\Sigma^{-1}|}\right).
\end{aligned} \tag{B.10}$$

Here, we still applying the matrix inversion lemma to (B.10), and the expression in numerator and denominator can be transformed to

$$\begin{aligned}
\Sigma_Z^{-1} - \Sigma_Z^{-1}\Gamma\Sigma_Z^{-1} &= \Sigma_Z^{-1} - \Sigma_Z^{-1}(D^{-1} + \Sigma_Z^{-1})^{-1}\Sigma_Z^{-1} \\
&= (D + \Sigma_Z)^{-1}
\end{aligned} \tag{B.11}$$

$$\begin{aligned}
\Sigma_Z^{-1} - \Sigma_Z^{-1}\Sigma\Sigma_Z^{-1} &= \Sigma_Z^{-1} - \Sigma_Z^{-1}(\sigma_X^{-1} + \Sigma_Z^{-1})^{-1}\Sigma_Z^{-1} \\
&= (\Sigma_X + \Sigma_Z)^{-1}.
\end{aligned} \tag{B.12}$$

Therefore, (B..10) can be derived to a further extent as

$$\begin{aligned}\frac{1}{2}\log\frac{|C\Sigma_X C^T+\Sigma_Q|}{|\Sigma_Q|} &= \frac{1}{2}\log\left(\frac{|\Sigma_Z^{-1}-\Sigma_Z^{-1}\Gamma\Sigma_Z^{-1}|\cdot|\Gamma^{-1}|}{|\Sigma_Z^{-1}-\Sigma_Z^{-1}\Sigma\Sigma_Z^{-1}|\cdot|\Sigma^{-1}|}\right)\\ &= \frac{1}{2}\log\frac{|D+\Sigma_Z|^{-1}\cdot|\Gamma|^{-1}}{|\Sigma_X+\Sigma_Z|^{-1}\cdot|\Sigma|^{-1}}\\ &= \frac{1}{2}\log\frac{|\Sigma_X+\Sigma_Z|\cdot|\Sigma|}{|D+\Sigma_Z|\cdot|\Gamma|}\\ &= \frac{1}{2}\log\frac{|\Sigma_X+\Sigma_Z||\mathrm{diag}(\Sigma_1,\cdots,\Sigma_L)|}{|D+\Sigma_Z||\mathrm{diag}(\Gamma_1,\cdots,\Gamma_L)|}.\end{aligned} \tag{B.13}$$

Bibliography

[1] W.Thomas, S.Heiko: Source coding: Part I of fundamentals of source and video coding. Found. Trends Signal Process **4** (2011) 1–222

[2] ITU-T, ISO/IEC: Lossless and near-lossless compression of continuoustone still images. ITU-T Rec. T.87 and ISO/IEC 14495-1 (JPEG-LS) (June, 1998)

[3] ITU-T, ISO/IEC: Digital compression and coding of continuous-tone still images. ITU-T Rec. T.81 and ISO/IEC 10918-1 (JPEG) (September, 1992)

[4] ITU-T, ISO/IEC: Advanced video coding for generic audiovisual services. ITU-T Rec. H.264 and ISO/IEC 14496-10 (MPEG-4 AVC) (March 2010)

[5] W.Thomas, S.Heiko: Source coding: Part I of fundamentals of source and video coding. Found. Trends Signal Process **4** (2011) 69–101

[6] J.Chen, L.Xie, Y.Chang, J.Wang, Y.Wang: Generalized Gaussian multiterminal source coding: The symmetric case. CoRR (2017)

[7] D.S.Slepian, J.K.Wolf: Noiseless coding of correlated information sources. IEEE Trans. Information Theory **19** (1973) 471–480

[8] A.D.Wyner, J.Ziv: The rate-distortion function for source coding with side information at the decoder. IEEE Trans. Inf. Theor. **22** (1976) 1–10

[9] S.Tung: Multiterminal source coding (ph.d. thesis abstr.). IEEE Trans. Inf. Theor. **24** (1978) 787

[10] J.Wang, J.Chen: VectorGaussian multiterminal source coding. IEEE Trans. Inf. Theor. **60** (2014) 5533–5552

[11] Y.Oohama: Rate-distortion theory for Gaussian multiterminal source coding systems with several side informations at the decoder. IEEE Trans. Inf. Theor. **51** (2005) 2577–2593

[12] J.Wang, J.Chen: On the sum rate of vector Gaussian multiterminal source coding. 2010 IEEE International Symposium on Information Theory (2010) 46–50

[13] Y.Oohama: Indirect and direct Gaussian distributed source coding problems. IEEE Trans. Inf. Theor. **60** (2014) 7506–7539

[14] Y.Yang, Y.Zhang, Z.Xiong: A new sufficient condition for sum-rate tightness in quadratic Gaussian multiterminal source coding. IEEE Trans. Inf. Theor. **59** (2013) 408–423

[15] J.Wang, J.Chen: Vector Gaussian two-terminal source coding. IEEE Trans. Inf. Theor. **59** (2013) 3693–3708

[16] M.Gastpar, P.L.Dragotti, M.Vetterli: The distributed Karhunen-Loève transform. IEEE Trans. Inf. Theor. **52** (2006)

[17] A.Wyner: The common information of two dependent random variables. IEEE Trans. Inf. Theor. **21**(2) (2006)

[18] M.A.Woodbury: Inverting modified matrices. Memorandum Rept. 42 (1950)