4-9-2020

Study of Fundamental Tradeoff Between Deliverable and Private Information in Statistical Inference

Farhang Bayat
Louisiana State University and Agricultural and Mechanical College

STUDY OF FUNDAMENTAL TRADEOFF BETWEEN DELIVERABLE AND PRIVATE INFORMATION IN STATISTICAL INFERENCE

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 Division of Electrical and Computer Engineering

by
Farhang Bayat
B.S. in Electrical Engineering, Amirkabir University of Technology, 2012
M.S., Amirkabir University of Technology, 2014
March 2020

Acknowledgments

This dissertation would not be possible without several contributions. It is a pleasure to thank the National Science Foundation who have supported me under Grant No. 1320351.

It is a pleasure also to thank Dr. Shuangqing Wei for providing me with a pleasant working environment where I felt comfortable presenting my views and he was always ready to help me in times of need.

This dissertation is dedicated to my family specifically my mother, father and sister Farimah, for their constant support and encouragement.

Table of Contents

List of Tables

List of Figures

Abstract

My primary objective in this dissertation is to establish a framework under which I launch a systematic study of the fundamental tradeoff between deliverable and private information in statistical inference. My research was partly motivated by arising and prevailing privacy concerns of users in many machine learning problems.

In this dissertation, I begin by introducing examples where I am concerned of privacy leakage versus decision utility in statistical inference problems. I then go into further details about what I have achieved in formulating and solving such problems using information theory related metrics in a variety of settings. Both related works and my own results are later summarized in the first chapter.

In the second chapter, I introduce a problem of detecting any subgraph using binary codeword queries. Furthermore, I seek and find limits imposed by the privacy of each graph which help me develop an understanding of privacy versus utility problems.

In the third chapter, I shift my focus from the original graphical framework to a more general bin allocation problem motivated by addressing concerns on privacy leakage in regard to users' web surfing patterns with usage of proxy or VPN services. After problem formulation, I deem it necessary to introduce submodular functions as a means of simplifying such problems and finding their solutions.

In chapter four, I expand upon the concept introduced in chapter three by allowing uncertainty between hypotheses and find the relationship between distinguishability, privacy leakage and utility in a deterministic bin allocation framework.

In chapters five and six, motivated by my previous works, I shift my focus to the problem of tradeoff between utility and leaked information when a randomization, rather than a deterministic mapping, is introduced as a privacy protection mechanism. In particular, I first seek solutions using a typical and widely accepted Information Bottleneck (IB) approach. I then detail how the original information bottleneck method does not necessarily provide an optimal solution to the proposed problem. I then offer my own novel approach based upon Augmented Lagrange Multipliers (ALM) and Alternating Direction Method of Multipliers (ADMM) with both theoretical justification and empirical evidence , as well as the inherent structures of both the objective function and privacy constraints. My approach has been shown to attain notable improvements than that under the IB framework, with well justified enhancement on efficiency of local convergence.

Finally in chapter seven, I present plans to cope with issues of lacking true statistics, by exploiting a set of information theoretical measures which have been shown to be equipped with more benign properties in robustness against limited amount of training data than the regular mutual information measure.

Chapter 1
Introduction

Throughout this chapter, I aim to offer insight as to the motivations of my current research in as detailed a manner as possible by offering everyday examples. Furthermore, a chronological summary of my work is provided. I go into details as to how every step of the way was first initiated and how it led to the next step so as to offer insight into the logical progress of the work. In the next step, I highlight my contributions so as to reiterate how my work is both novel and valuable. Following is a section dedicated to the related works I have taken advantage of to some degree in the development of my results. Finally, I will finish this chapter by detailing the organization of the rest of this dissertation.

1.1 Motivation

Automated decision making systems based on statistical inference and learning are widely used in real-world applications today. These systems have an initial input of a dataset containing many items and their corresponding features and the decisions made for such items. Then they use machine learning to come up with an underlying algorithm which given the same features could result in the same decisions for these inputs. These algorithms are designed over the features of such inputs. By this method, the system has managed to develop an in-built predictor which now has the ability to make decisions for a new set of inputs based on their features. Furthermore, such a system's predictor could always be updated with new inputs thus staying current. However, there are still some detriments to such systems worthy of notice.

All of these systems are trained based on historical data. This in term means that the system is designed to honor the previous set of decisions for every input regardless of whether such decisions were fair or not. Rather, these systems choose to blindly accept previous decisions made upon the data and simply try and follow them through. Thus, if there ever were any kind of injustice in the original process of decision making, such unfair behavior will never be noted or fixed. Such unfair data could result in discrimination, which is defined as gratuitous distinction between individuals with different sensitive attributes. This in turn results in a continuous unfair behavior in the automated decision making system which will also influence future behavior. It is important to note that although such a scenario might seem far-fetched, they happen more often than one could suspect and the reason behind such unfairness usually lays in historical or cultural differences between past and present time.

We thus introduce a new goal; to design the same automated decision-making systems that could help us predict any decision for future data while limiting the amount of information revealed about the user's sensitive features. As a direct result, I clearly cannot use the same dataset as before since it reveals some sensitive information. But I still need a clean version of this data to make any kind of inference system. Thus a tradeoff problem blossoms.

In this dissertation, I set my goal as finding a tradeoff between deliverable information (the information required for making a decision with little to do with sensitive information) and private information (the sensitive information which might result in unfair treatment of a group of users). In order to better understand the predicament and my role in this tradeoff problem, I offer a number of examples:

Example 1: One of the most sensitive features of any person in today's world is their race. As an example, in the United States, courts use features of criminals such as their age, race, sex, years of being in jail and so forth to estimate their possible recidivism (future arrest)[1]. Then based on these estimates, a judge chooses whether to set a prisoner free or not. While such a system might seem quite fair at first, there is an underlying discrimination at play.

In the not so distant past and due to cultural climates, there used to be a tendency for police officers to treat African-Americans more harshly and thus more African-Americans tended to be arrested again in the future. Thus, in the data currently present at the hands of the court, there is a bias towards setting African-American prisoners free, simply because the data from the past suggest that they have a higher likelihood of being arrested again. However, missing from this data is the fact that most of these arrests were more than likely just a product of another time and culture. Thus judging new prisoners based on such criteria would be considered discrimination.

Example 2: Another sensitive attribute in a person in today's world is their gender. To find examples of a bias in gender representations I can simply look at some image search results in any online search engine [2]. In a Google image search for “CEO”, a mere 11 percent of the depicted results are women while a whopping 89 percent are men. These results are at odds with reality seeing as how in the real world, 27 percent of U.S. chef executives are women. As another example, in a Google image search for “telemarketer”, 64 percent of the depicted results are women while in the real-world the male to female ratio in telemarketing positions is almost 50-50.

Once again, these results show an underlying discrimination towards women and their position in today's world. Google search results are a collection of data (in this case images) gathered through the years. Now since the positions of "CEO" and "telemarketer" used to be very male and female-oriented respectively, does not mean that image results should be more male or female-oriented as well. Rather, a fair search result should be able to demonstrate the reality of the world today rather than be affected by previous matters.

Solution: Both these examples showcase a less-explored side of privacy problems. In such problems, I am still trying to develop a predictor. However, private attributes such as race and gender need to be kept either completely hidden or revealed to a minimum. Furthermore, I cannot completely dismiss the previous decisions either. Rather, whatever predictor I develop, it still needs to be offering decisions quite close to the previous predictors whether it was the likelihood of a prisoner to be arrested again or the image of a CEO or telemarketer. I thus choose to design my decision making system through the disclosable section of the data and try and cut off all connections with the private section. I note that there might be interconnections between the sensitive and disclosable features (like a prisoner's race and their residential Zip code) and offer algorithms on how to deal with this problem as well. Furthermore, I choose an auxiliary variable to design an optimal compression channel helping us reduce the number of features (otherwise if all the features could be saved, I might have over-fitting rendering the final results pointless)

Throughout this discussion, I assume that (1) the true distribution of the private features is known and (2) the true statistical interconnection between private and non-private features is known. Later on, in Chapter 7, I offer insight on how to deal with cases where I only have access to empirical versions of these two statistics.

1.2 Research Tasks Completed

1.2.1 Detection of Hidden Active Subgraphs

The process of formulating such questions and offering solutions to them started when I posed a question of the average time it would take to detect a hidden subgraph in a larger network of graphs. Such a question was first imposed in other works such as [3, 4]. However, in their research the authors assumed a very strict characteristic on the subgraphs they aimed to detect such as the subgraph being two colorable or complete respectively. This assumption prompted us to ask the same question for any possible set of subgraphs as long as they were edge-wise disjointed. If solved, the solution could help us find further insight into questions of community detection within a higher level network where the edges within the same subgraph showcase a stronger connection compared with those placed outside the subgraph. For example, a group of people on Facebook might be more deeply connected to one another based on their taste in musical artists in comparison with another group who might enjoy a different set of artists. The goal would then be to detect which of these groups is active at a certain time in the shortest time possible by following a set of general queries (such as age, gender, etc). After doing so, the detector is able to introduce these people to one another as a commodity (like the friend or page suggestions on Facebook). Furthermore, the detector is able to offer these users well-suited advertisements based upon their portfolio. However, he/she is not able to further separate these networks to offer more specifically-suited ads to each member within the subgraph.

Assuming that the queries were made in the form of bit allocations and offering a definition of feasible queries -so as to make certain of distinguishability of different edgewise disjoint subgraphs- I was able to offer algorithms on how I could carry out such a task. Introducing the concept of query(attribute) tables, I was able to find decision trees for every desirable subgraph. I then offered insight as to how these decision trees could work to my advantage to help find an average stopping time for each subgraph. Finally, by carrying out a second level average stopping time over the sum of all subgraphs I was able to find the average time it takes to detect the first of such subgraphs (the active subgraph). The results of this section of the research were finalized and published in [5].

First, I proposed a novel framework to study the problem of sequential detection of active substructures under the constraint of protection of edge-wise activity patterns. I began by offering a definition of privacy within this framework as a means to a better interpretation of the constraints concerning my discussion. I then formulated the novel problem of detecting active subgraphs from a given set of link-wise disjoint substructures. I showed how such active graphs could be identified by querying with a series of feasible binary queries satisfying the constraint of protecting link-wise states over the detection period, a feat whose representation is further interpreted using vertex covering of subgraphs. Furthermore, I introduced the relationship between partial and complete vertex covering and the resulting breach of privacy imposed by the latter in my problem. A random coding approach was proposed to establish a sequential and non-adaptive binary search process whose average stopping time is analyzed based on both upper and lower bounds. Then the complexity of the method was calculated and shown to be efficient. Finally, simulation results were provided to demonstrate the efficiency of proposed bounds.

This is where I introduced the concept of utility versus privacy which formed the basis for the rest of my research. To do so, I introduced the definition of privacy in my framework and showcased how it changes to utility function. The results of this section of my work were gathered and published in [6]. A culmination of my work thus far has been presented in Chapter 2 which mostly covers my journal paper [6] since [5] is simply a precursor to the former.

1.2.2 Utility versus Privacy

Once I was introduced to the concept of privacy in such networks, I shifted my focus towards privacy in communication networks and other utilities which could be developed over such a background. A significant example could be found in personal security versus meta data gathered by different agencies. Meta data refers to the overall information which could be gathered about each of us as a collection of our online browse history is gathered over a week, a month or even longer. As mentioned beforehand, the use of internet is so intertwined with our lives these days that abandoning it is not an option. While there are still methods of hiding your online activities, they all have a certain cost. The best tool to fight leaking information would be the use of a Virtual Private Network (VPN) which would change a user's IP address to a virtual (or at least untraceable) IP thus rendering the user untraceable. However, the use of a VPN slows the connection down thus resulting in a loss of bitrate.

Another method of fighting such leaks is the use of proxy websites. These sites also change the user's IP address so as to cover his/her trace and they also slow down the connection. However, a user is allowed to utilize more than one of them at the same time allowing him/her to divide the bitrate load over more than one network and thus possibly achieving a higher bitrate utility. I acknowledge that using more than one proxy site is going to result in a higher privacy leakage but still deem it worthwhile in comparison with a lesser loss of bandwidth.

To model this solution I introduce a binning problem where the goal is to browse a total of M desired websites each with a certain frequency using N proxy sites. my goal is then to hide the frequency of each site use. A service provider usually traces the browsing history of a user by following his/her URLs. It follows that if $N = 1$, all the websites are visited through the same proxy and thus no information about the frequency of either of them is revealed. On the other hand if $N = M$, then each website is visited through one proxy and thus all the browsing frequency information is revealed. Since I am attempting to find a balance between these two extremes I assume $N < M$ and try to find the best binning (allocation) algorithm to maximize the overall utility described as a mixture of bitrate utility and leaked information.

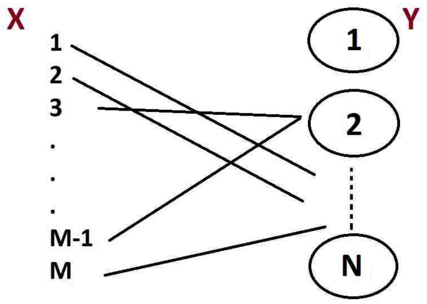

FIGURE 1.1. Binning problem Representation of M-to-N proxy use

To solve such a problem I find it necessary to introduce submodular and subsequently multi-submodular functions which I aim to use to find a solution for my proposed problem. Due to the complex nature of submodular functions, I am required to find a series of sufficient conditions for my utility function which guarantees the existence of a solution for $2 < N < M$ but unfortunately does not offer a closed form solution. Then for the case of $N = 2$ I offer both the same and more simplified sets of sufficient conditions for the existence of a solution. Furthermore, for this case, I offer algorithms as to how the problem could be fully solved using algorithms previously developed. The results of this section were submitted and recently accepted to the 2018 International Conference on Telecommunications [7] and are represented in Chapter 3 of this dissertation.

1.2.3 Hypotheses Detection Under Privacy Constraints

A natural progress from my previous work would be to aim the question of what if a user wants to be distinguishable from other users so as to enjoy the accommodations offered by service provider to individuals. As an example, they wish to be offered a series of well-suited suggestions on Amazon while not being tracked on every items they may view. In other words, the user aims to be detected as an individual but not have his/her vital information revealed. For such a goal the user could once again choose applying multiple proxies for different browses. By doing so the user (1) hides a portion of his browsing information by applying it through the same proxy thus rendering it indistinguishable (2) leaves a portion of information open to interpretation as a means of detection.

Rather, the user marks a group of browses as the same, thus imposing ambiguity among them and offers a chance to the service provider to detect him/her as an individual by the frequency of each proxy's use and corresponding output.

Thus if a utility function based upon distinguishability between a number of users is calculable, a privacy constrained problem between the user and an eavesdropper (for example a service provider) could be defined. I further assume the existence of a cost function imposed by the use of any proxy websites (assuming they might charge the user for their imposed load). The solution to such a problem could offer insight in regards to the tradeoff between proxy allocation utility and meta information leakage when I face the problems of partitioning one of multiple possible sets of random items (i.e. websites that 2 or more users have chosen to visit following their own distributions) into a given number of bins with the hopes of distinguishing the users (i.e. a given set of proxy servers each of which has its own cost function). The results of this study were recently submitted to IEEE Transactions on Information Forensics and Security [8] and are also presented in Chapter 4 of the dissertation.

1.2.4 Revisiting Information Bottleneck

Until this section of my research, I was mainly concerned with finding the optimal deterministic channel to accommodate different privacy and hypotheses distinguishability goals. Then I decided to look into the design a probabilistic channel for the same goals. To do so, I revisited the original Information Bottleneck problem [9] and the solution offered for it. I then discovered the many inadequacies of this problem's solution which had not only to do with the non-convexity of the problem but also with imposing Bayes rule upon the variables ,based solely on following the path of [10] and [11] and with the one goal of a simple answer. I thus found it interesting to utilize recent mathematical approaches to finding a solution for a non-convex optimization problem. To do so, I first used the Augmented Lagrange Multipliers (ALM) method [12] to rewrite the constraints as part of the objective function. Then I introduced and used Alternating Direction Method of Multipliers (ADMM) [13] to find a more optimal solution to the original Information Bottleneck problem. I then showcased the superiority of my method in comparison with those of the original IB solution through numerical results.

The results of this study were submitted to and published in 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton) [14]. They can also be found in Chapter 5 of this dissertation.

1.2.5 Tradeoff between Disclosable and Private Latent Information revealed via Compressed One: an ADMM-Based Approach

Motivated by my previous work on the original Information Bottleneck problem, I attempted to offer a formulation of a more complicated problem. Here, I described the goal as finding the optimal tradeoff between disclosable and private latent information through a compression channel. Whereas in Chapter 5 mutual information was chosen as a measure of revealed information due to the nature of the Markov chain in the compression network, here I offer my own reasoning behind the use of mutual information as a measure of revealed information using the concept of typical sequences in a consecutively used communication channel; thus rendering such a measure novel. Although the final formulation of the problem turns out to be quite close to that of [15], it is important to note that (1) in [15], it was simply assumed that mutual information would be an acceptable measure of revealed information (2) in [15], the final solution to the problem was just an expansion on the original information bottleneck problem wherein the simplicity of the solution is preferred over how well it performs as long as there's a brief mention of how the problem at hand is a non-convex constrained optimization problem. However, due to my experience, I am equipped with a new set of tools, namely Augmented Lagrange Multipliers (ALM) and Alternating Direction Method of Multipliers (ADMM) which I utilize in two different ways to offer a solution to the formulated problem. I then use numerical results to indicate how my method offers a better solution than simply implementing the Information Bottleneck method as suggested by [15].

The results of this study are in the process of submission to IEEE Transactions on Information Forensics and Security and can also be found in Chapter 6 of this dissertation.

1.2.6 Empirical Data Analysis of the Same Problem

After an overall review of the contributions so far in the field of privacy and utility, I find it important to note how all the analysis made until now was based on the assumption that I have access to the true distribution of the data whether it be the condition channel between input and median in Chapter 5 or the interconnected channels between disclosable and private inputs and the median in Chapter 6. However, this assumption is almost never true. In the real-world, I only have access to a limited number of samples from these two channels and am forced to assume an empirical distribution between the two of them. Then the question becomes "Are I certain that the designed system based on empirical data will also be compatible with true data?". Rather, is there a guarantee that once a system is designed for the optimal tradeoff based on a limited number of samples, will it still be any good for an unlimited number of samples or will it begin to falter? A response to this question was proposed in [16]; where it was deduced that for a specific set of privacy measures such a hypotheses will hold true. Unfortunately, in the same paper, it was shown that mutual information is not one of such privacy measures, seeing as how it is not Lipschitz continuous. This thus results in a push for us to change my measure of privacy from mutual information to another (compatible) measure and revise all previous results accordingly. This represents my current ongoing work a better understanding of which has been presented in Chapter 7.

1.3 Summary of Contributions

Throughout my research, I was able to impose many problems as models of real life situations and offer solution algorithms and insight into them. Following is an extensive list of my contributions through this dissertation:

In Chapter 2, I

(1) expand upon the ideas of random group testing and hidden graph detection to find an average stopping time for the detection of any set of edge-wise disjoint subgraph using a codeword based sets of queries and introducing decision tables and trees to carry out such a goal.

(2) introduce the concept of feasibility in any graph detection problem and then go into further details as to what the existence or lack of such a constraint imposes upon the network

(3) introduce the concept of utility versus privacy in the framework of hidden graph detection and discussing the trade-off between these two as both an additional point to the hidden graph detection problem and a step towards the second half of my research where privacy becomes a main point of concern.

In Chapter 3, I

(4) offer insight into issues of meta data and its corresponding privacy constraints and how the solution to it could be modeled in the form of a binning problem where each bin represents a possible proxy website

(5) introduce submodular and multi-submodular functions as the required tools for solving the problem of avoiding privacy leakage and their corresponding representation in the form of diminishing returns property

(6) find the set of sufficient conditions for the existence of a solution to the problem offered and modeled in the previous two steps for a general N number of possible bins (proxy sites) and then offering less restrictive sets of sufficient conditions for the existence of a solution to the problem under the assumption that $N = 2$ and offering a solution algorithm by the end of which the optimal bin allocation scheme is well-defined.

In Chapter 4, I

(7) introduce a measure of distinguishability between 2 hypothesis plus a measure of average leaked information given any number of the hypothesis is active;

(8) develop a formulation of a multi-agent multi-variant optimization problem with a privacy leakage constraint

(9) offer insight into the complexity of such an NP-hard problem and further sufficient conditions under which I can simplify it into a polynomial problem

(10) offer a description of the algorithm utilized to find the solution given the sufficient conditions followed by the proximity results and resort to numerical exam-

ples to further demonstrate the applicability of submodular solutions, as compared with the results using exhaustive search in manageable settings.

In Chapter 5, I

(11) revisit the Information Bottleneck problem and the algorithm to solving it as suggested by [9] and explain exactly if and how any step in the original algorithm could be problematic

(12) introduce the concept of ALM and ADMM solutions to the same original problem and showcase every possible superiority offered by the new method of looking at the problem

(13) offer an algorithm and an in-depth look at how it could be implemented to my problem and demonstrate the practical results of running my suggested algorithm (ADMM) over the same problem; and offer in-depth reasoning for the numerical results to further justify the novelty and importance of my new suggestion.

In Chapter 6, I

(14) offer a novel formulation of the problem of tradeoff between disclosable and private information through a compressed one with a novel justification of the use of mutual information as a measure of privacy

(15) go into details as to why the information bottleneck-based method of solving such a problem is not necessarily optimal by going through every step in detail

(16) utilize the concept of ALM and ADMM solutions for the new problem and showcase every possible superiority offered by the new method of looking at the problem

(17) offer two separate ADMM-based algorithms and an in-depth look at how they could be implemented to my problem and demonstrate the practical results of running these suggested algorithms (ADMM) over the same problem; and offer

in-depth reasoning for the numerical results to further justify the novelty and importance of my new suggestions.

Finally, in Chapter 7, I

(18) introduce the main issue with any of my suggested methods of privacy so far which lays in the empirical nature of real data in any machine learning problem

(19) offer some insight as to how such issues could be addressed thus paving the way for future studies in the field.

1.4 Related Works

In this section I would like to go into details as to which papers and how each of them helped us develop my conceptual understanding of the problem.

1.4.1 Non-Adaptive Sequential Detection of Active Edge-Wise Disjoint Subgraphs Under Privacy Constraints

Our interest in hidden subgraph detection came from previous works in group testing where the main goal was the detection of a number of defective items between a large set in as short of a time span as possible. Examples of such utilization could be further witnessed in the fields of disease diagnosis, experimental design and active learning [17, 18], to name a few.

In [3, 19] a similar issue was raised where the objective was to find a lower bound on time required to establish any two-colorable graph. However, my work assumes a set of particular candidates upon the detection of which a decision should be made rather than just a two-colorability condition. Also note that in [3, 19] it was assumed that only one edge is causing the actual failure, however in my case any of the edges within a certain component could lead the system astray. Thus, my only utilization of [3, 19] comes down to borrowing the idea of node and edge removals based on random coding.

Unlike group testing problems where a certain actual active set shall be identified [20], the final graph in my work has multiple equivalent candidates. Compared to the problems tackled in [3, 19], both encoding and decoding in this discussion have to be done sequentially rather than blockwise -where a lower bound was offered. More importantly, a particular labeled graph out of M candidates has to be built through communications, rather than an arbitrary two colorable graph without any other specification. Thus, both the codebook and decoding have to be tailored to this purpose. And the primary goal is to maximize the efficiency of building the desired graph in terms of finding an optimal probabilistic encoding and corresponding decoding strategies to attain the minimum average decision stopping time at which the graph is formed.

It should be noted that sequential group testing under graph constraints has been considered in [21] where tested nodes must form a path in a given graph [22]. However, in my settings, there is no such restriction on tested nodes in each time slot. Rather, the set of active node pairs are selected from one of M candidate subgraphs.

[23] also dealt with a similar situation where emphasis was put on finding structures where some fault may be hidden. However in their respective work, they did not assume any known substructures and tested every circuit for a possible error and then forming a respective component based on these search results. Thus, in their model system case, all edges are observable. However in my model, I am searching for a hidden network which can only be probed and studied based on a codebook and a binary channel output vector Y.

After finishing my first publication [5], I aimed to expand upon my previous studies by finding more real life examples where my design of the problem could be applied to interesting results.

We first came across some studies concerned with understanding community structures within multiple types of networks. In [24], an example of such nature concerning social networks was offered. Most problems associated with this field are closely tied with the concept of community detection. Community detection refers to the problem of clustering nodes [24] or links [25] in a given network (which structure-wise is represented through a graph) into multiple substructures under which the infrastructure connectivity density is higher than that between different substructures. Borrowing this concept, I opt to shift my focus towards the issue of detecting active substructures assuming I already have access to a list of such possible edge-wise disjoint subgraphs or community ties using any existing link-wise community detection algorithm. [26, 27, 28].

The reason why I choose edge-wise disjoint subgraphs rather than vertex-wise disjoint subgraphs is due to how overlapping vertexes could be utilized in characterizing relationships where agents may get involved within multiple contexts, e.g friendship networks, collaboration networks [25, 26, 27]. After identifying such subgraphs, my goal is to find an active subgraph by utilizing queries about structure-wise activity states without compromising private information of actual link-wise states. In that sense, my work could be considered along the line of active learning but with constraints of protecting local private information in a network [29, 30].

In all previous cases of detection I already had access to a prior set of queries and hypothesis and aimed to differentiate the latter in the form of a set of known tests and outcomes [31, 32, 33, 34]. However due to my new constraints, I need to first find a set of feasible queries. Furthermore, I need to establish the relationship between query codewords and subgraphs in terms of a query table in accordance with the properties of a given set of edge-wise disjoint subgraphs, given a priori in my problem. In other words, I must offer a method of building a table indicating the relationship between such tests and outcomes as an added contribution. Then, sequential querying process could be carried out and the insight their corresponding outputs offer will help indicate which substructure is active after a certain number of observations.

In my work, I may face multiple issues such as possibly large size networks, potential feedback noise in responses to the queries as well as subgraph deconstructions. To handle such problems, I opt for a random query method (a more information theoretically inspired approach) best known as the random coding method [35, 36], to seek both the upper and lower bounds on the average stopping time of the proposed problem.

As mentioned previously, my proposed framework for active graph detection could find connections to existing works on group testing and detection of some specific active non-labelled structures. Group testing as introduced in [37] could be graphically modeled as starting with a graph $G(V, E)$ where $|V| = N$ and $|E| = \binom{N}{2}$ with the goal of cutting down G in as short a time span as possible to end up with $G(V', E')$ where $|V'| = k$ and $|E'| = \binom{k}{2}$. This was carried out by sequentially removing k_k subgraphs (complete subgraphs of order k) or unions of star shaped trees. In [4], the goal was changed to detecting graphs of a certain characteristic (complete or star shaped) by allowing the graphs to have overlapping edges. Queries were made in the form of complete graphs and whether or not they shared any edges with the graph I was hoping to detect. Following the same concept was [3] where the authors attempted to find a stopping point where $G(V', E')$ contains only two defective nodes and is also a two colorable graph rather than a graph of a certain order or a certain shape.

1.4.2 Partition of Random Items: Tradeoff between Binning Utility and Meta Information Leakage

In my proposed framework, meta data information refers to the patterns about a sequence of items(for example the user's favorable websites) infer-able based upon a sequence of bins (e.g. proxy sites) observed by an eavesdropper. This assumption is an expansion of [38] and [39].

The concept of privacy has already been explored in many works such as [40, 41] where a general but non-mathematical explanation was offered. However; in my work, I go into further details as to what privacy represents in my framework and how it could be formulated into many settings. Later, I found it necessary to utilize the concept of multi-submodular set function problems and their solutions. This concept was widely discussed in [42] where they introduced a series of sufficient conditions on multi-submodular set functions by which the multi-submodular problem could be transformed into a submodular set function problem. Then, further discussions about the existence of a solution to the new problem were made. By doing so -and if a solution was proven to exist-, the complexity of the problem could be shown to be reduced from NP to polynomial. However, [42] did not offer any algorithmic solutions in such cases which unfortunately results in us simply proving the existence of a solution rather offering an algorithm to support such solution as well. Thus, I simply utilize the works of [42] to define problems who should have a solution.

1.4.3 Partition of Random Items: Tradeoff between Binning Utility, Meta Information Leakage and Hypotheses Distinguishability

To the best of my knowledge, the closest work to mine was done by [43]. In [43], they also considered the tradeoff between distinguishability and information leakage where the former is quantified using KL-distance and the latter using mutual information. However, in [43], (1) they assumed a probabilistic mapping between inputs and outputs, but I am considering deterministic and singular mapping (i.e. many-to-one); and (2) they did not consider costs associated with such mapping. In addition, in order to reduce the complexity of their problem from NP to P, they further pursued special cases with negligible information leakage. On the other hand, in my problem I relax the objective to seek conditions for submodularity structure in my discrete optimization framework.

It later turns out that in order to simplify the solution complexities of the problem, I impose a quadratic property over the cost function. However, as is further explained in [44], such an assumption is not necessarily as limiting as it seems, seeing as how in many economic models, functions are written as extensions of quadratic functions.

It is important to note that one of the most prominent measures of privacy as described in works such as [45] has been differential privacy. Differential privacy is mainly concerned with limiting the information leaked through different sequences generated by a randomized process. In my problem settings, differential privacy aims to investigate the effects of a minimal change in the input in the overall output of the system. In other words, if there are originally X^n sequences at my disposal which are mapped to Y^n sequences in the output, is there a minor change in the input sequence that could result in an out of control change in the output sequence?

Overall, differential privacy in literature is concerned of one-shot measure ([46]). However, all my measures are in average sense including average utility, mutual information, KL divergence, whose functional significance rests upon repeated drawing from a distribution over long run. Thus differential privacy is not of concern here. Even if I could consider differential privacy in the settings of long sequences, such privacy measure is not quite relevant due to the long-sequence constraints.

As for the addition of distinguishability, to the best of my knowledge, the closest work to mine was done by [47] where they considered the tradeoff between distinguishability and information leakage when the former is quantified using KL-distance and the latter using mutual information. However, they formulated the problem using the concept of missed detection and false alarm as the basis of hypotheses testing. In this work, the authors were interested in a randomized binning process which tended to be more complex than my routine. Such a goal came at the cost of sacrificing the concept of trade-off by only allowing very small privacy leakage (less than ϵ) in order to invoke a first order approximation to reduce the computational complexity of the optimization problems. However; in my work, I am concerned with any amount of trade-off with deterministic binning to seek suboptimal and polynomial order approximations enabled by the fundamental properties of multi-submodular set functions.

The multi-submodular function was discussed extensively in [42] where they introduced a series of sufficient conditions on multi-submodular set functions by which the multi-submodular problem could be transformed into a submodular problem. Then, further discussions about the existence of a solution to the new problem were made. By doing so -and if a solution were proven to exist-, the complexity of the problem could be shown to be reduced from NP to polynomial. Such sufficient conditions have been adopted by us in in seeking proper binning utility functions under which such sufficient conditions hold true.

1.4.4 Information Bottleneck Problem Revisited

Due to the nature of my chapter, which heavily relies on discussing every step of the information bottleneck method, my main point of reference will be the original Information Bottleneck paper [9]. I later find it important to note that [9] was inspired by the works of [10] and [11]. Thus, from time to time, in order to indicate the inadequacies of [9]'s results, I will refer to these papers. Further comments on the inadequacies of [9] have been made in many works as recent as [15] which reflect the concerns of many previous authors who while aware of the limitations of IB, still chose to utilize it due to its simple implementation.

The idea of solving Lagrange multiplier problems where the goal is to either maximize or minimize a function using penalty functions has been explored in many previous mathematical and computing works such as [12]. These issues could be modeled in the form of an Augmented Lagrange Multiplier (ALM) problem. However, due to the still-complex nature of solving any non-convex Lagrange multiplier problem (which requires carrying out gradient descent and checking for conditions on every Lagrange multiplier), further studies and methods are required. One algorithm of dealing with the limitations of an ALM problem is to instead try and adopt an ADMM algorithm. One of the most recent works about this method is [13] which details the superiority of adopting ADMM over continuing with ALM. [13] makes further observations on the accuracy of a non-convex Augmented Lagrange Multiplier problem which I will utilize to justify the numerical results achieved through this chapter.

Finally, [48] offers a series of sufficient conditions on the desired utility function by which a convergence of the ADMM method could be guaranteed. I will offer insight into these sets of sufficient conditions and whether or not they are applicable to my problem settings.

1.4.5 Tradeoff between Disclosable and Private Latent Information revealed via Compressed One: an ADMM-Based Approach

We recently became aware of the works done in [15] which closely resemble my overall formulation of the problem. However, in my work, my definition of the original problem and how I come to formulate it as such are quite different. Furthermore, even though information bottleneck alike approach has also been considered by us as [15], it should be noted this approach serves as a comparison reference, leading to my novel new algorithm. . Also of note is how [15] did not offer any numerical results for the method developed and simply sufficed to reiterate how the problem is non-convex (as was the original IB problem) and that the results are not necessarily optimal. While it is true that the overall problem is still non-convex, as I will show, there are methods that could outdo simply running the Information Bottleneck method again with little fixtures. I will detail such novel approaches and algorithms whose performances are compared with the IB based approach using numerical results.

In order to introduce my method, I find it necessary to introduce the concept of Augmented Lagrange Multiplier (ALM) [12] and later on Alternating Direction Multiplier Method (ADMM) [13]. Overall, this chapter is a further generalization of my previous work in [14] where I discussed applying ADMM to the original information bottleneck problem; however without an additional privacy constraint.

1.4.6 Proposed Works: Empirical Data Analysis

Seeing as how in Chapter 7, I am mainly concerned with the implications of using empirical data in my problems, my main source of study has been [16] where I use the works done to develop some ideas as to how I could generalize my previous results.

1.5 Dissertation Organization

In Chapter 2, I will present my journal paper [6] where the main goal is to find the average time it would take to detect a hidden subgraph and form an opinion on the relationship between the elapsed time and the privacy leakage. Then in Chapter 3, I offer the full length version of my conference paper [7] a shorter version of which was made to match the page limitations of the conference requirements [14]. In this section, the goal is to generalize the utility function hinted at in Chapter 2 to find a more general formulation of the problem concerning utility versus privacy. In Chapter 4, another level of uncertainty is added to the problem formulated in Chapter 3 where I aim to also distinguish different possible hypotheses while maximizing utility and keeping the leaked information to a constraint. This in turn, results in a journal paper [8] fully presented in Chapter 4. In chapter 5, I present my conference paper [14] where as a natural progression from Chapter 4, I look at the problem of finding an optimal probabilistic channel between input and output for a good tradeoff between utility and privacy. I thus come across the concept of the Information Bottleneck problem and its solution [9]. I offer insight about the pros and cons of this method and then offer my own method based upon ALM and ADMM to find better results than those gathered in [9]. In Chapter 6, I offer my current journal paper in progress where introduce the concept of private and disclosable information and offer a formulation of the previous privacy-utility optimization problem where different sections of information are meant to be hidden or revealed. I then use the previously discussed ADMM approaches to find two new solutions and compare them with those of using the original Information Bottleneck method. Finally, in Chapter 7, I discuss a few issues with my assumptions in problem formulation so far and offer insight on how I can deal with such shortcomings in future works.

Chapter 2

Non-Adaptive Sequential Detection of Active Edge-Wise Disjoint Subgraphs Under Privacy Constraints

2.1 Introduction

In my work, I provide an algorithm to find average stopping time for any feasible graph with no specifications on either the shape or the order of the final graph. This then results in calculation complexities higher than those offered by previous works. Furthermore I need to impose edge overlapping or non-edge-overlapping characteristics to my problem and am thus able to offer more insight into the issues of privacy. The privacy issue studied herein is due to possible revelation about edge activity patterns of a given active sub-graph, which can be inferred based on query outputs. Such information leakage is further quantified using average mutual information between edge status and query outputs.

Overall, my work in [5] mainly dealt with finding an average stopping time for detecting an active hidden subgraph given enough feasible queries and primarily focused on the mathematical aspect of such a detection problem. In this discussion, I address the possible issue of an insufficient number of feasible codewords and how to deal with it, the consequent tradeoff between privacy and subgraph detection and all new detection complexities. More specifically, the main contributions of this discussion are listed as follows: (1) development of algorithms to evaluate average stopping time for any tree shaped decision intersection hypergraph; (2) finding the calculation complexities faced when using the random query method; (3) defining and evaluation of privacy in terms of information leakage on edge activity patterns. The privacy issue further prompts us to propose an expansive notion of feasible queries, as well as intersection hyper graphs for the detection purpose. The resulting tradeoff between average stopping time, and information leakage is then further evaluated.

The rest of the chapter is organized as follows. I present my system model and problem formulation in Section 2.2. Then in Section 2.3, I provide the basis of privacy within my framework of study and later rationalize why feasibility is an issue which needs to be addressed. In Section 2.4, I offer insights as to whether such problems are always solvable; that is whether I am able to distinguish a given set of subgraphs under the constraint of protected edge-wise activity information. In Section 2.5, I offer a mathematical solution to the problem I have raised using Bayes' decision rule. In Section 2.6, by adapting the proposed random coding approach, I find upper and lower bounds for the average stopping time of the formulated sequential and non-adaptive subgraph detection problem and compare the final results. Simulation results are also demonstrated in Section 2.7 to assist any further understanding. Furthermore, I offer calculation complexity levels for all solutions. Finally, in Section 2.8 I offer a comparison between my approach and the deterministic non-adaptive algorithm and show it is not an acceptable method for my problem.

2.2 System Model and Formulation

In my model I assume there are M edge-wise disjoint subgraphs $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2), ...G_M = (V_M, E_M)$, where V_i and E_i denote the vertex and edge sets for G_i. There are in total N vertices in these subgraphs, and each vertex could belong to multiple structures. Over the course of my test, only one of these subgraphs is active, for example G_1. The state of G_1 being active is equivalent to stating that each edge in E_i is equally likely to be active at each testing time slot.

Over each testing time slot, a codeword (i.e. a query) of N binary bits is used to inquire the status of the hidden subgraph. The relationship between the allocation of N bits and the response of the subgraph G_i to the query given that G_i is the hidden active subgraph, shall follow these rules : (1) If vertex covering imposed by the nodes allocated with 1 bits covers all the edges of a subgraph G_i, the output of the system is 1 ; (2) If vertex covering imposed by the nodes assigned with 1 bits covers none of the edges of subgraph G_i, the output of the system is 0. Since codewords are at my disposal, I will make certain only codewords resulting in one of the 2 above situations are allowed. This measure is taken as a means of protecting privacy in metadata as further discussed in Section 2.3.

During each time slot, all subgraphs whose outputs are different from the one observed are removed from the list of candidate subgraphs. As a result of a sequence of such queries and candid subgraph removals, the desired subgraph is found. It follows that the stopping time is subject to both the set of queries used and the actual active subgraph. My first objective is to find a feasible set $A = \{\mathbf{a_1}, \mathbf{a_2}, ...\mathbf{a_K}\}$ of codewords where $\mathbf{a_j} \in \{0,1\}^N$ for $1 \leq j \leq K$, subsets of whom result in the detection of each edge-wise disjoint G_i subgraphs. This means that every G_i could be constructed using one of i_{L_i} (referred to from now on as decision factors) sets gathered in $S_i = \{S_{i_1}, S_{i_2}, ..., S_{i_{L_i}}\}$ where (a) $S_{i_k} \subseteq A$ and (b) $S_{i_k} \not\subseteq S_{i_l}$ where $k \neq l$ and $k, l \in \{1, 2, ..., L_i\}$. For the non-adaptive sequential detection problem, my goal is to develop algorithms to find an acceptable sequence of codewords from the set A to run a sequence of Boolean queries whose outcomes enable us to determine which subgraph is being active, with a minimum average stopping time. The average is taken over the prior distribution of the states of the given M subgraphs. Without loss of generality, I assume all subgraphs are equally likely to be active in the rest of the chapter.

2.3 The Issue of Privacy

In this section, I aim to quantize privacy in a more strict manner so as to deal with it in a mathematical framework.

To do so, assuming I have identified the active subgraph G_i I would like to find more information about each of the edges within this subgraph and their (supposed) probability of activity. I allow a new definition of partial covering stating that the output is 1 if and only if the active edge is adjacent to at least one node with bit 1 and the output is 0 otherwise. Then, if I repeat the same query many times, a probability distribution of activity over such edges is developed meaning the uncertainty among these edges becomes less and thus new insight is developed.

In other words, a codeword **a**, could be modeled as a channel which will divide the edges of an active graph G_i into two sets; those with output 0 labelled $E[0]$ and those with output 1 labelled $E[1]$. This could be modeled in Figure 2.1 where the probability distribution of each such sets could be calculated as $\pi_{E[0]} = \sum_{e \in E[0]} \pi_e$ and $\pi_{E[1]} = \sum_{e \in E[1]} \pi_e$ where π_x is the probability distribution function of variable x and the index $e \in E[G_i]$.

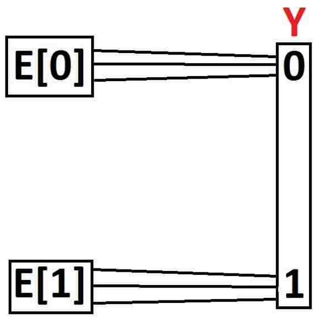

FIGURE 2.1. Communication Channel Representation of Codeword **a**

Now, I would like to see how much information about the inputs could be revealed using the binary output $Y \in \{0, 1\}$. Assuming $S_e = E[0] \cup E[1]$ and X numerically representing each member of S_e, I can have:

$$I[X;Y|\mathbf{a},G_i] = H[Y|\mathbf{a},G_i] - H[Y|X,\mathbf{a},G_i] = H[Y|\mathbf{a},G_i] \tag{2.1}$$

In the last equation I have used the fact that knowing $\mathbf{a}$, G_i and X will let us know exactly what Y is going to be and thus $H[Y|X,\mathbf{a},G_i] = 0$.

I also know that Y follows the probability distribution of $P[Y=0] = \pi_{E[0]} = 1 - P[Y=1]$. Thus the mutual information between X and Y will be equal to

$$I[X;Y|\mathbf{a},G_i] = H[Y|\mathbf{a},G_i] = h\left(\pi_{E[0]}\right) \tag{2.2}$$

where I have introduced $h(q) = -q\log(q) - (1-q)\log(1-q)$, where $q \in [0,1]$. The above equation dictates that if codeword $\mathbf{a}$ was repeatedly chosen as a channel to transfer bits of information over, based on outputs 1 and 0, a certain level of information would be revealed about edge variations and thus assumptions about the original edges could have been made. Then if n channel uses were made (a query a is used n times repeatedly), at most $2^{nI(X;Y)}$ number of messages each of which represents a particular edge-wise activity pattern can be revealed based on the vector of $[Y_1, Y_2, \cdots, Y_n]$. In other words, mutual information $I\left(X;Y|\mathbf{a},G_i\right)$ quantifies the rate at which information in regard to edge-wise activity patterns is disclosed due to adoption of a query codeword $\mathbf{a}$ which offers a partial vertex covering for the given graph G_i.

It then follows that for the codeword $\mathbf{a} = [a_1, a_2, ..., a_N]'$ and based on the number of edges these N bits cover, the above mutual information could vary between 0 and 1 bit.

As an example, I assume the graph in Figure 2.2 represents an active subgraph. Under this framework, if I assume p_{ij} represents the probability of the edge between nodes i and j being active, then all I know is that $p_{12} + p_{23} + p_{34} + p_{41} = 1$.

Now, I opt to apply a certain codeword to such a graph.

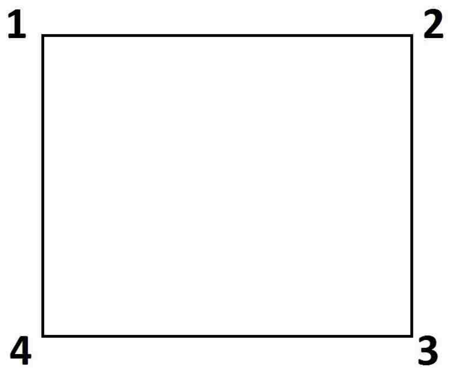

FIGURE 2.2. Active Subgraph Example

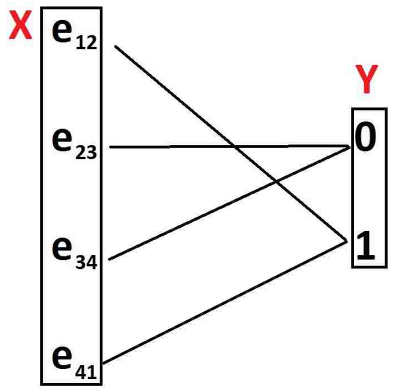

FIGURE 2.3. Mapping Possibly active edges to Outputs Given $\mathbf{a_1}$

An example could be viewed in Fig. 2.3 where codeword $\mathbf{a_1} = [1000]'$ has been applied to each (possibly active) edge and the resulting outputs have been witnessed. Here, I am able to write a formula representing the mutual information within original inputs which models the relationship between the active edges and the observed outputs given codeword $\mathbf{a_1}$ was chosen.

$$I(Y;X|\mathbf{a_1}) = H(Y|\mathbf{a_1}) \tag{2.3}$$

where for this specific example $H(Y|\mathbf{a_1}) = h(p_{12}+p_{41})$ represents the entropy over the final output.

It thus becomes obvious that by choosing a codeword and running it repeatedly, I will be able to obtain new information about the edge-activity patterns. It is important to note that some codewords will not be able to offer us any new information such as codeword $\mathbf{a_2} = [1010]'$ for which I will have:

$$I(Y;X|C_1) = H(Y|\mathbf{a_2}) =$$

$$-(p_{12} + p_{41} + p_{23} + p_{34})\log(p_{12} + p_{41}p_{23} + p_{34}) = 0 \tag{2.4}$$

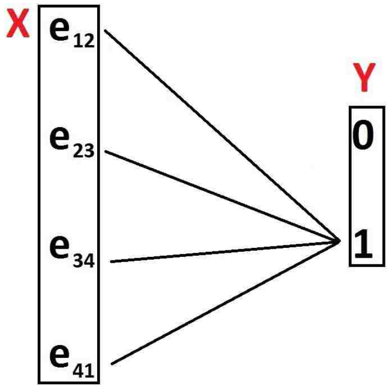

FIGURE 2.4. Mapping Possibly active edges to Outputs Given $\mathbf{a_2}$

where I have used the fact that $p_{12} + p_{23} + p_{34} + p_{41} = 1$.

Now I am able to define invasion of privacy in my framework of graphical structures. If by the use of certain codeword a_j, further information about the patterns of active edges within an active subgraph G_i is revealed, I concur a breach of privacy has occurred which can be quantized by the formula $BoP_{i,j} = I(X;Y|\mathbf{a_j}, G_i)$ where BoP represents Breach of Privacy.

Generalizing the above formula over a number of codewords a_j each with probability $P(a_j)$ and possible active subgraphs G_i each with probability $P(G_i)$ of occurrence will result in the following formulation for a measure of privacy invasion which quantifies the average amount of information leaked about the edge activity patterns of a detected graph G_i among M candidates.

$$BoP(\mathbf{G}) = \sum_{1}^{M} P(G_i) \sum_{j=1}^{i_{L_i}} P(a_j) BoP_{i,j} \tag{2.5}$$

2.4 Queries and Codewords

In a binary setting, a codeword over original graph $G = (V, E)$ where $|V| = N$ could have 1 of 2^N possible forms. Thus in the first step I am able to form an $M \times 2^N$ table T where every cell T_{ij} represents the output of subgraph G_i, $1 \leq i \leq M$ after applying codeword $\mathbf{a_j}, 1 \leq j \leq 2^N$.

Following from my ideas in Section 2.3, I search for any cell T_{ij} whose output is neither 0 nor 1. Such a cell shows that $\mathbf{a_j}$ is causing inconsistency for subgraph G_i and is thus not a feasible codeword. After a thorough search over all $M \times 2^N$ cells, I am able to find all such codewords which cause inconsistency in even 1 respective subgraph and remove them. The table remaining will have a size of $M \times K'$ where K' represents the number of consistent codewords. All such problems are in relation to the concept of complexity in search for vertex covering which [50] shows is an NP-complete problem. Tables 2.1 and 2.2 represent the initial binary response relationship between queries and outputs for detection of the set of graphs in Fig. 2.5 and Fig. 2.6, respectively.

2.4.1 Detectability of Subgraphs with Feasible Queries

It's important to note that in [4, 3] (a) the subgraphs were not labelled and (b) subgraphs were not assumed to be of a certain shape and size. In my work, I need to first determine if a given set of edge disjoint subgraphs can be distinguished using consistent vector queries.

I am adapting sequential detection which means the results of my previous observations will help us limit the next set of results. This, in turn, means that those series of 1s and 0s which offer the same output for all possible subgraphs are idle to detection; thus a number of consistent codewords should be removed.

I anticipate that the set of acceptable codewords or queries might not be sufficient to let us differentiate all given subgraphs. I illustrate this issue using two example

TABLE 2.1. Outputs of each subgraph to each codeword in Figure 2.5

	$\mathbf{b_1}$	$\mathbf{b_2}$	$\mathbf{b_3}$	$\mathbf{b_4}$
G_1	1	1	1	1
G_2	0	1	1	1
G_3	1	1	1	1
G_4	1	1	1	1
G_5	1	1	0	1
G_6	1	0	1	1
G_7	1	1	0	0

where one set of $M = 7$ of $N = 6$ nodes is given in Figure 2.5 and another set of $M = 5$ of $N = 6$ nodes is given in Figure 2.6. In the first case, the set of acceptable codewords consists of $\mathbf{b_1} = [000111]'$, $\mathbf{b_2} = [110101]'$, $\mathbf{b_3} = [111000]'$, and $\mathbf{b_4} = [111100]'$ while for the second case there is only one accpetable codeword $\mathbf{d_1} = [110010]'$, both of which, as explained next, turn out to be not enough.

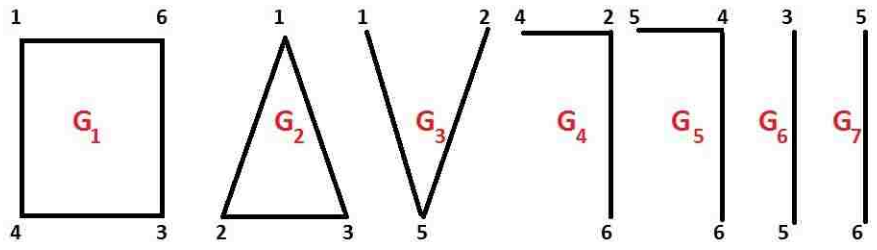

FIGURE 2.5. Original Subgraphs of Example 1

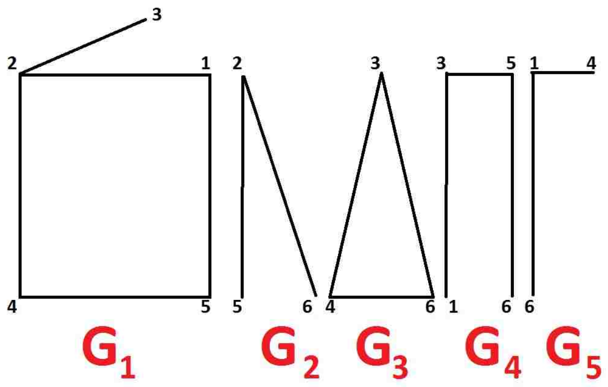

FIGURE 2.6. Original Subgraphs of Example 2

The reason why I am able to say that the number of codewords for Figures 2.5 and 2.6 is not enough could be seen in both Tables 2.1 and 2.2; in Table 2.1 if the

TABLE 2.2. Outputs of each subgraph to each codeword in Figure 2.6

	$\mathbf{d_1}$
G_1	1
G_2	1
G_3	0
G_4	1
G_5	1

active subgraph is any of G_1, G_3 or G_4, there is no distinction between them which could be detected by the selection of codewords I have. The same similarity is seen between G_1, G_2, G_4 and G_5 in Table 2.2 when I suffice to d_1.

Thus I should add new codewords (queries) to my current selection. However; these codewords should still hold up the consistency requirement of the model. It then follows that to find such codewords I need to change the shapes of some subgraphs.

I want to find the most number of new applicable codewords with minimum number of changes. Thus, I need to find the general form of the codewords consistent with all subgraphs. Usually, the subgraph with maximum number of edges imposes the most consistency issues since there is a higher possibility of inconsistent edges within such subgraphs. So, I try to find the general form of the codeword consistent with the subgraph with maximum number of edges. To do so, I must make sure that no two adjacent nodes within such subgraphs are allocated the same bit. This is because if such same bits were both 0s, they could result in a 0 edge within the subgraph and thus cause inconsistency. Once the general form of such codewords has been found, I need to make sure these codewords are also consistent with all other subgraphs. Thus, I apply this codeword form to all remaining subgraphs; if any two adjacent nodes in the following subgraphs are offered the same bit, I remove the edge within them from mentioned subgraph and treat it as a single edge subgraph. Once all such single edge subgraphs have been found, I have found a form of codewords which is compatible with all new subgraphs. If the number of newly added subgraphs is not enough, I will follow the same steps as before this time starting from the subgraph with second most number of edges and so forth until enough codewords have been gathered.

To model the general form of consistent codewords, I suggest using graph coloring. After coloring a certain subgraph G all nodes within G will have a different color than their adjacent nodes. Thus if a certain node is given color r (which represents a 0 bit in my framework) I can make sure that all adjacent nodes are given contrasting colors b, g,... all of which represent a 1 bit in my framework. Furthermore, if a certain node is given color b (which represents a 1 bit in my framework) all adjacent nodes are free to choose whichever color they want as long as there is no inconsistency induced amongst them.

Now, I am able to introduce the same method using vertex coloring frameworks. Assuming there are M_L subgraphs, $|1s|_L$ single edge graphs and C_L applicable codewords in the L^{th} step of the algorithm, the algorithm could be summed up as:

Algorithm 1: Algorithm to generate enough codewords for detection given a set of initial subgraphs

```
0. $L = 1$;
1. Find the subgraph $G_i$ with maximum number of edges.
2. Color the $G_i$.
3. Remove $G_i$ from the list of possible subgraphs.
4. Treat all nodes with the same color as single edge subgraphs.
5. Check to see if $2^{|C_L|-|1s_L|} \geq M - |1s_L|$. If yes go to 6, otherwise, go to 1.
6. End
```

The stopping condition results from the fact that every single edge subgraph could be detected using one and only one codeword (which consists of all 1s and two 0s in the place of adjacent nodes of the edge). This means that if there are M_L

TABLE 2.3. Outputs of each subgraph to each codeword in Figure 2.7

	$\mathbf{a_1}$	$\mathbf{a_2}$	$\mathbf{a_3}$	$\mathbf{a_4}$	$\mathbf{a_5}$	$\mathbf{a_6}$	$\mathbf{a_7}$
G_1	0	1	1	1	1	1	1
G_2	0	0	1	1	1	1	1
G_3	1	1	1	1	1	1	1
G_4	0	1	0	1	1	1	1
G_{51}	1	1	1	1	0	0	1
G_{52}	0	1	0	1	0	1	0
G_6	1	1	1	0	1	1	1
G_7	1	1	1	1	0	1	1

TABLE 2.4. Outputs of each subgraph to each codeword in Figure 2.8

	$\mathbf{f_3}$	$\mathbf{f_4}$	$\mathbf{f_5}$	$\mathbf{f_6}$	$\mathbf{f_7}$	$\mathbf{f_8}$	$\mathbf{f_9}$	$\mathbf{f_{10}}$
G_1	1	1	1	1	1	1	1	1
G_2	1	1	1	0	1	1	1	1
G_{31}	1	1	1	1	0	1	1	1
G_{32}	1	1	1	1	0	0	1	1
G_{411}	1	1	1	1	1	1	0	1
G_{412}	1	1	1	0	1	1	1	0
G_{42}	0	1	1	1	1	1	1	1
G_{51}	1	1	0	1	1	1	1	1
G_{52}	1	0	1	1	1	1	1	1

subgraphs and $|1s|_L$ single edge subgraphs, $|1s|_L$ codewords will be used to simply identify the single edge subgraphs. Thus if there are a total of $|C_L|$ codewords available, seeing as how there are only 2 possible outputs sequentially speaking, I at least need to satisfy the above inequality wherein the left side shows the number of possible different subgraphs.

This algorithm results in the original set of subgraphs in Figure 2.5 to change into the new set visible in Figure 2.7. This example took 1 repetition to find the final set of subgraphs and its final codewords are $\mathbf{a_1} = [000010]', \mathbf{a_2} = [000111]', \mathbf{a_3} = [101010]', \mathbf{a_4} = [110101]', \mathbf{a_5} = [111000]', \mathbf{a_6} = [111001]', \mathbf{a_7} = [111010]', \mathbf{a_8} = [111100]'$. The same algorithm for the original set of subgraphs in Figure 2.6 resulted in the set of graphs apparent in Figure 2.8 and the corresponding codewords $\mathbf{f_1} = [010010]', \mathbf{f_2} = [010011]', \mathbf{f_3} = [010111]', \mathbf{f_4} = [011011]', \mathbf{f_5} = [011110]'$ and $\mathbf{f_6} =$

$[101100]', \mathbf{f_7} = [110010]', \mathbf{f_8} = [110011]', \mathbf{f_9} = [110101]', \mathbf{f_{10}} = [111100]'$. These codewords took 3 repetitions to develop. I see that in both cases $2^{|C_L|-|1s_L|} > M - |1s_L|$ meaning I have more than satisfied the necessary condition and thus one or more of the codewords could be dropped with no detection problems arising, and thus I drop $\mathbf{a_8}$ from the first and $\mathbf{f_1}$ and $\mathbf{f_2}$ from the second case.

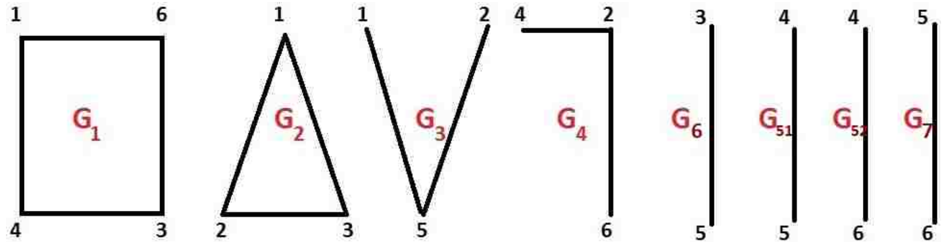

FIGURE 2.7. Final Subgraphs of Example 1

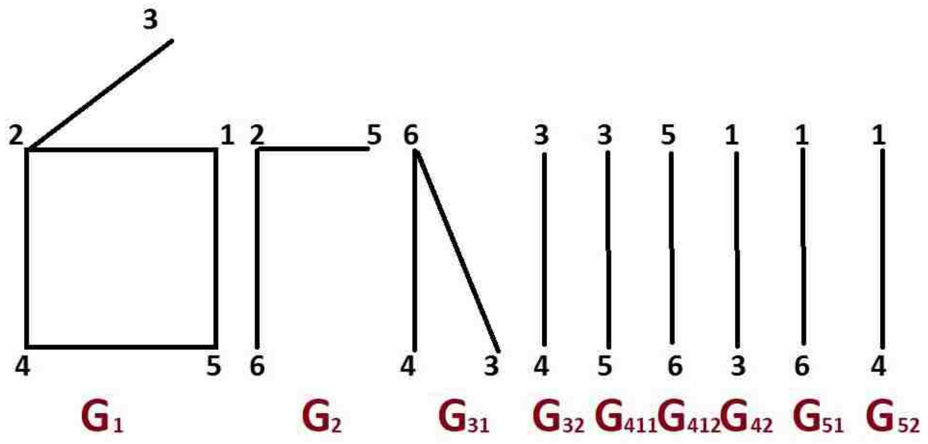

FIGURE 2.8. Final Subgraphs of Example 2

Thus by using the algorithm, I am able find the minimum number of changes I need to impose on the subgraphs to find enough codewords to solve the initial problem. However, by assuming new single edge subgraphs and new resulting subgraphs, I am compromising the level of privacy, as described in Section 2.3.

Still, given the assumption that all edges within an active subgraph are equally likely to be active, I am able to use these minimum changes to find an average stopping time for the original subgraphs. As an example, I can find the average stopping time of G_5 in Figure 2.5 by calculating the average stopping times of G_{51} and G_{52} in Figure 2.7 and then averaging the result over the number of edges within the two new subgraphs meaning $E(\lambda_{G_{5,old}}) = 0.5E(\lambda_{G_{51,new}}) + 0.5E(\lambda_{G_{52,new}})$ where λ indicates the stopping time variable. Furthermore in Figure 2.8 I will have

$$E(\lambda_{G_{3,old}}) = \frac{2}{3}E(\lambda_{G_{31,new}}) + \frac{1}{3}E(\lambda_{G_{32,new}}) \tag{2.6}$$

2.4.2 Methods of Codebook Construction

For implementation of such a network, I have chosen a probabilistic approach, where I assume that at every time slot one of the codewords in A is randomly chosen to transmit upon. Using these codewords, the detector performs a semi generalized binary search, which will end up with forming the chosen subgraph. THere, I can assume a binomial distribution for each codeword $\mathbf{a_j} \in \{0,1\}^N$ for $1 \leq j \leq K$ meaning $P(\mathbf{a_j}) = p^{N(1s)}(1-p)^{N-N(1s)}$ where $0 \leq p \leq 1$ and $N(1s)$ represents the number of 1 bits in codeword $\mathbf{a_j}$ and then find the optimal p^* under which I generate the codewords randomly following the assigned probability distribution. As long as the average stopping time over all graphs can be minimized, there must exist a good sequential codebook to achieve such performance. I will go into further details as to why this method is optimal in Section 2.8.

2.5 Solution

In this section, I offer a method that helps solve the problem introduced in Section 2.2. These methods are based upon random codebook detection as explained in Section 2.4.2. I will offer further reasoning and analysis hidden behind this method and specified algorithms concerning it. I will start my work by decision factor identification and then move on to the random code generation method.

2.5.1 Decision Factor Identification

In this section, I explain how decision factors for a certain subgraph are discovered. In order to do so, I would advise the reader to review [31] where the main goal was to find a sequence of queries to form the most symmetric decision tree possible for a set of queries and hypotheses. [31] could then be modified to help us find the most effective decision factors for a certain subgraph.

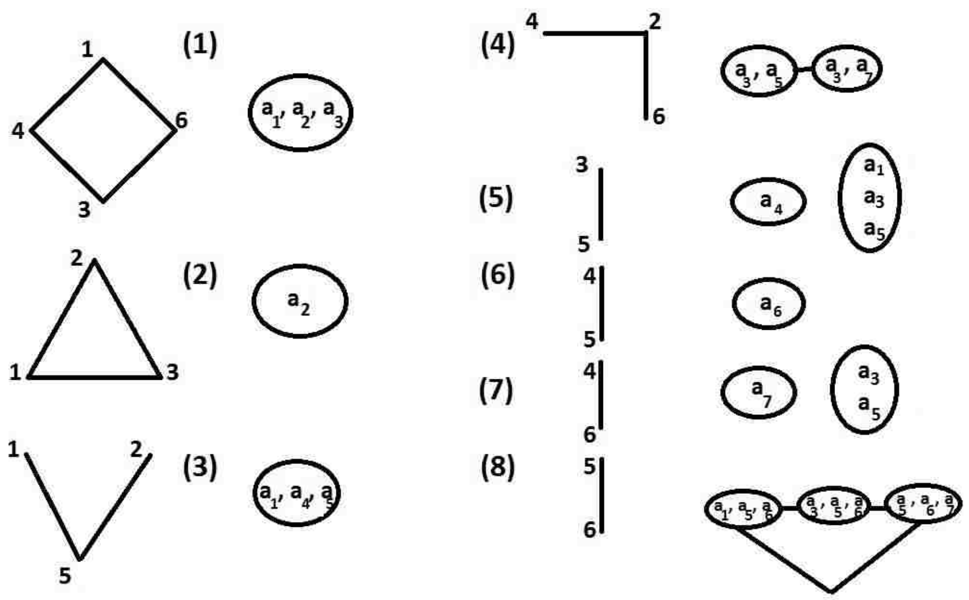

FIGURE 2.9. Decision Factors for Example in Figure 2.7

The most effective decision factors distinguish the target subgraph as soon as possible; thus if I were to label the system's output to subgraph $G_i, i = 1, ..., M$ and codeword $\mathbf{a_k}$, $k = 1, ..., |C|$ as $T_{i,k}$, then I would have access to all possible $M \times |C|$ outputs as witnessed in Table 2.1.

Since I am attempting to find the fastest stopping time factors, it would be logical that for a given active subgraph G_i, I would choose the codeword $C_{i,1}$ which puts it in the most distinguished selection of subgraphs where $(i, 1)$ index represents the number 1 choice of a codeword for detection of active graph G_i. By doing so repeatedly, I will be able to fully distinguish G_i from all other possible choices with the minimum number of queries.

The idea presented above represents a greedy method of finding the best possible solution factors. However, such an algorithm might only offer a local optimum point for my solution. This is because at every step I choose the best option, I am choosing one locally optimum choice and following it with another. It may in return be possible that I am missing an overall better representation due to such strict local choices. Thus, I add a randomized selection method as well to account for such cases where in every step top 10 percent of the codeword choices are accounted for as possible solutions and thus an overall group of possible solutions is developed.

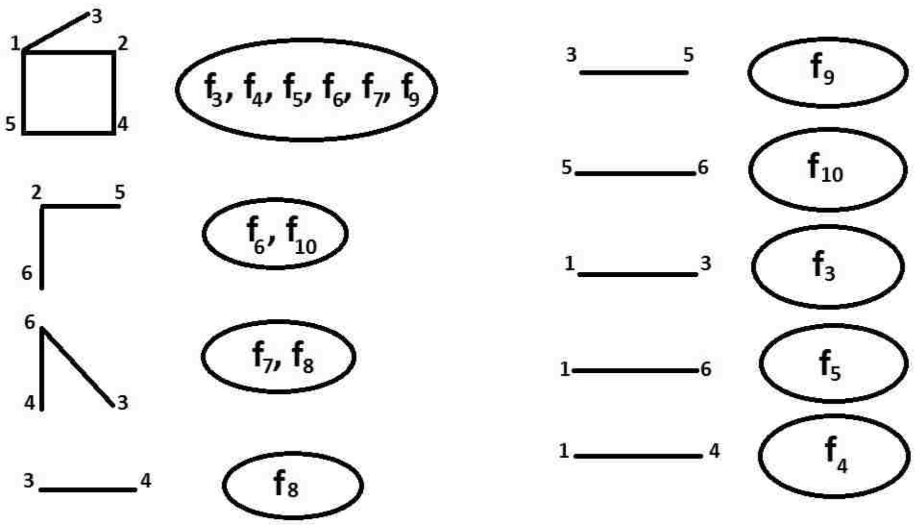

FIGURE 2.10. Decision Factors for Example in Figure 2.8

Algorithm 2: Randomized Decision Factor Identification Algorithm Given a Set of Subgraphs and Codewords

```
1. Choose the target subgraph $G_j$

2. Find the codewords $c_{k'}$ such that the number of subgraphs with the same
   output to the codeword $c_{k'}$ as those of $G_j$ would be in the minimum 10
   percent
   Repeat the following steps for each of these codewords
```

```
        i. If there are more than 1 such codewords, choose each of them separately
        ii. Write down $c_{k'}$ as the first decision factor
    3. Form a new output matrix consisting of only those subgraphs with the same output to the codeword $c_{k'}$ as that of $G_j$.
    4. If the size of the output matrix is not $1 \times 1$, go to 2 Otherwise, go to 5
    5. End.
```

Applying the algorithm above will result in the best decision factors for every subgraph in Fig. 2.7 and 2.8 to be as observed in Fig. 2.9 and 2.10 respectively. I plot the results as hypergraphs where each node represents a series of codewords and two nodes are adjacent if they share at least one codeword. I call such hypergraphs intersection graphs.

It is important to note that Algorithm 2 is basically offering a decision tree whose every vertex is a codeword. Union of these codewords on every walk from the root to any of the branches then represents a decision factor. It follows that any intersection graph will be in the form of a complete graph since every vertex (decision factor) at least shares the root with all other vertices. I now aim to calculate the maximum number of vertices on an intersection graph.

I know that the number of branches in a binary decision tree is maximal when all the vertices have the lowest degree. The vertices cannot all have degree 0, because such a condition would assume that at least two different codewords could identify the same subgraph alone which means the codewords must represent complementary vertex coverings. However, such coverings do not occur in my binary framework.

It thus follows that the number of branches in a binary decision tree is maximal when all the vertices have degree 1 except one vertex. In such a case, if there are $|C|$ codewords, there could be a total of $|C|-1$ branches. Then the intersection graph will form a $|C|-1$ complete graph (as witnessed in Fig. 2.11). Thus, the

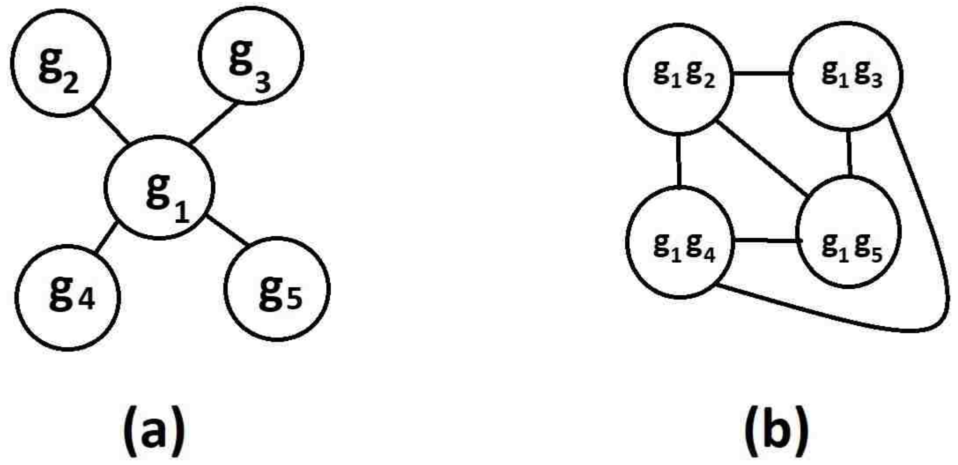

FIGURE 2.11. Example on two representations of decision factors

most number of vertices an intersection graph for the detection of a subgraph G_i could have is equal to $|C|-1$. It then follows that for a second, third and fourth subgraph $G_j, j \neq i$ this number is equal to a maximum of $|C|-1$ since the same decision factor (such as $\{\mathbf{g_1g_2}\}$ in Fig. 2.11) could reoccur as long as the outputs to the same query are different (for example when the output given $\mathbf{g_1g_2}$ and subgraphs $G_j, G_{j+1}, G_{j+2}, G_{j+3}$ active, is $[00], [01], [11], [10]$ respectively). However for the fifth subgraph this maximum is dropped by 1 since the same decision factor cannot reappear. It then follows that for the detection of M subgraphs the maximum number of vertices in all decision factors will be equal to:

$$4(|C|-1+|C|-2+...+|C|-\frac{M}{4}) = \\ M|C|-\frac{M}{2}(\frac{M}{4}+1) \tag{2.7}$$

I will use Eq. (2.7) to calculate the complexity of the solution for my problem.

2.5.2 Implanting the Bayes Optimization Rule

I see that there are 2 possible types of decision factors: (a) they each consist of several combinations of $S_{i_l}, 1 \leq l \leq L_i$ (as introduced in Section 2.2) which do not share any $\mathbf{a_e}$ with each other (b) they consist of several combinations of S_{i_l} which have some $\mathbf{a_e}$ in common with one another. For a better understanding, in Figure 2.9 subgraphs (4) and (8) fall into category (b) and all the rest follow the definition of category (a).

For the first case, in order to compute the probability distribution of the overall stopping time, I would simply calculate the stopping time of each possible S_{i_l} and then use the Bayes rule as a means of determining which of them has occurred first. In other words I utilize the conditional probability mass function (PMF) to describe the stopping time given that G_i is active:

$$\begin{gathered} P[\lambda_{G_i} = n] = \\ P[\lambda_{S_{i_1}} = n] P[\lambda_{S_{i_2}} > n, \lambda_{S_{i_3}} > n, ... | \lambda_{S_{i_1}} = n] + ... \\ P[\lambda_{S_{i_{L_i}}} = n] P[\lambda_{S_{i_1}} > n, \lambda_{S_{i_2}} > n, ... | \lambda_{S_{i_1}} = n] \end{gathered} \tag{2.8}$$

where $P(\lambda_{G_i} = n)$ represents the probability of G_i being detected at time slot n and $P[\lambda_{S_{i_l}} = n]$ represents the probability of G_i being detected at time slot n by S_{i_l}. It would then follow that for case (a) I will have the above equation simplified to:

$$P[\lambda_{G_i} = n] = P[\lambda_{S_{i_1}} = n] ... P[\lambda_{S_{i_{L_i}}} > n] + ... \tag{2.9}$$

Next, all I need to do is find the probability distribution of stopping time caused by a group of exclusive queries presented in the form of $P[\lambda_{S_{i_j}} = n]$ in the above section. I am going to do so for the case of two or more of such queries occurring. I will go into details about one important case and then find a method to generalize my results to any number of agents.

Let us assume that G_i depends on two queries $\mathbf{a_1}$ and $\mathbf{a_2}$ both occurring. As with most probability distributions I start by writing down the cumulative probability distribution (CDF) of $x' = \{\mathbf{a_1}, \mathbf{a_2}\}$ simply as $P(\lambda_{G_i} \leq n) = P(\lambda_{\mathbf{a_1},\mathbf{a_2}} \leq n)$ where the index defines the decision factor detection is based upon and n represents a discrete variable defined for the number of time slots. Now I may continue:

$$\begin{aligned} P(\lambda_{G_i} \leq n) = P(\lambda_{\mathbf{a_1},\mathbf{a_2}} \leq n) = \\ P(\lambda_{\mathbf{a_1}} \leq n, \lambda_{\mathbf{a_2}} \leq n) = P(\lambda_{\mathbf{a_1}} \leq n)P(\lambda_{\mathbf{a_2}} \leq n) \\ -\sum_{i=1}^{n} P(\lambda_{\mathbf{a_1}} = i)P(\lambda_{\mathbf{a_2}} = i) \end{aligned} \tag{2.10}$$

where in the last line of formula, the first part is due to the exclusivity of individual actions and the second part is due to the fact that at any time slot only one of these two agents may occur and thus I cross out the possibility of double counting.

Now that I have managed to find the CDF of stopping time for G_i, I could quite easily find the PMF:

$$P(\lambda_{G_i} = n) = P(\lambda_{\mathbf{a_1},\mathbf{a_2}} \leq n) - P(\lambda_{\mathbf{a_1},\mathbf{a_2}} \leq n-1) \tag{2.11}$$

where I already know that $P(\lambda_{\mathbf{a_1}} \leq n) = 1 - (1 - P(\mathbf{a_1}))^{\mathbf{n}}$ which is the CDF of a geometric distribution.

So far, I have managed to find the formula for PMF of an event's stopping time when it consists of two exclusive agents happening. I can see that for any general number L of actions the CDF of G_i will contain, one initial $P(\lambda_{\mathbf{a_1}} \leq n)P(\lambda_{\mathbf{a_2}} \leq n)...P(\lambda_{\mathbf{a_L}} \leq n)$ plus a series of sums and deductions represented in the form of $\mathbf{a_1} \cup \mathbf{a_2} \cup ... \cup \mathbf{a_L}$. Thus, the derivation in decision factors of type (a) is complete and I have managed to find an analytical method to compute the CDF of the stopping time; and for the PMF I will add $P(\lambda_{S_{i_j}} = n) = P(\lambda_{S_{i_j}} \leq n) - P(\lambda_{S_{i_j}} \leq n-1)$.

2.5.3 Approximating the Average Stopping Time of Connected Sets

I now study the second case where possible solution combinations share one or more actions with one another. In such the intersection graph could either contain a cycle or it may simply be a hypertree. I will first deal with the issue of the tree and then cope with the cyclic case.

Any of the nodes in the tree could have caused the formation of the subgraph G_i; this is why once again, I will follow the same route as the one in case (a) with Equation 2.8 meaning I will calculate the probability of each of these combinations resulting in the stopping time excluding all others from stopping the network at any time before then. However, I cannot simplify the formulas as easily as the previous cases seeing as how they are connected to one another through the edges of a tree and thus affect conditional probabilities. Instead I can use the following algorithm:

```
Algorithm 3: Approximating the Average Stopping Time Part 1
  for every node $j$ where $1 \leq j \leq L_i$ where $L_i$ represents the number of such
possible sets

   0. Label all nodes as Child with counter $l_k$, $1 \leq k \leq L_i$.

   1. Choose node $j$ as Parent. Make $l_j \leftarrow 0$.

   2. If node $j$ is Parent send message "I'm Parent" to all neighbors along with
      $l_j$

   3. If node $k$ is labelled as Parent and receives "I'm Parent" do nothing.

   4. If node $k$ is labelled as Child and receives "I'm Parent" from node, mark
      $k$ as Parent and $j$, $l_k = l_j + 1$.
```

```
    5. go back to step 2 until no Child labels left and let $L \leftarrow l_k$
```

The above steps help us organize the tree in such a manner that the parent node will have a counter $l_j = 0$ and the nodes at a distance d from the root node will have a counter $l_k = d$. An example of such algorithm is provided in Figure 2.12 where I finally have $L = 3$. I then define $PMF(S_{i_{l_j=0}})$ as the probability

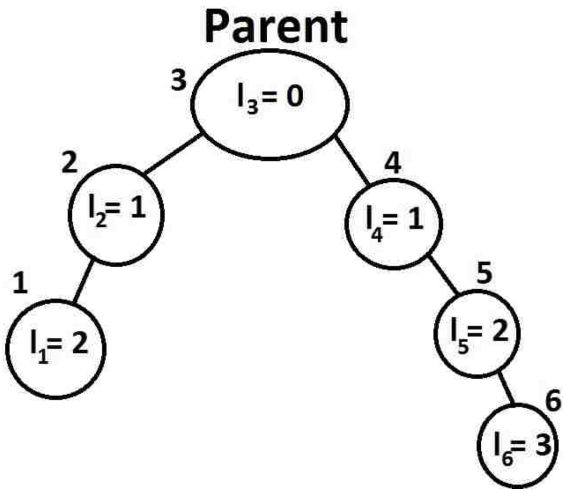

FIGURE 2.12. Output of Algorithm.3 with node 3 as parent

distribution function of stopping time for graph G_i given node j in G_i's decision factor graph is the stopping factor and $CDF(S_{i_{l_j}} - S_{i_{l'_j=l_j-1}}))$ as the cumulative density function of all agents in node j which are not appearing in node $j' = j - 1$. I can then offer the below algorithm to approximate the PMF of stopping time caused by node p:

```
Algorithm 4: Approximating the Average Stopping Time Part 2
  for every node $j$ where $1 \leq j \leq L_i$ where $L_i$ represents the number of such possible sets

    6. $PMF_{T_i,j} = 1$ and counter $\leftarrow 0$

    7. if $l_j = 0$ calculate the PMF of $j$ titled $PMF(S_{i_{l_j=0}})$ and $PMF_{T_i,j} \leftarrow PMF_{T_i,j} \times PMF(S_{i_{l_j=0}})$ and counter $\leftarrow$ counter $+ 1$
```

```
8. if $l_j > 0$ then $PMF_{T_i,j} \leftarrow PMF_{T_i,j} \times (1 - CDF(S_{i_{l_j}} - S_{i_{l'_j = l_j - 1}}))$ and
   $counter \leftarrow counter + 1$

9. $j \leftarrow j + 1$ and repeat 8 until $counter = L$
```

Thus, once I have attained all such probability distribution functions, the summation of them for all possible n will again offer us an amount which is proportional to the overall probability distribution function that is $PMF_{T_i} = \sum_{j=1}^{L_i} PMF_{T_{i,j}}$.

2.6 Cyclic Intersection Graphs: Upper and Lower Bounds

So far, I have provided an algorithm to deal with tree shaped intersection graphs. However adjusting this algorithm for cyclic intersection graphs proves to be complexity wise exponentially expensive. Instead, I aim to turn an intersection graph with cycles into a tree shaped intersection graph; I acknowledge the fact that changing the shape of the graph in any manner would cause my results not to be exact anymore and am thus going to use the idea of turning the shape with cycles into a tree as a means of finding upper and lower bounds for the average stopping time instead of exact results. To do so, I next introduce the concept of factor graphs.

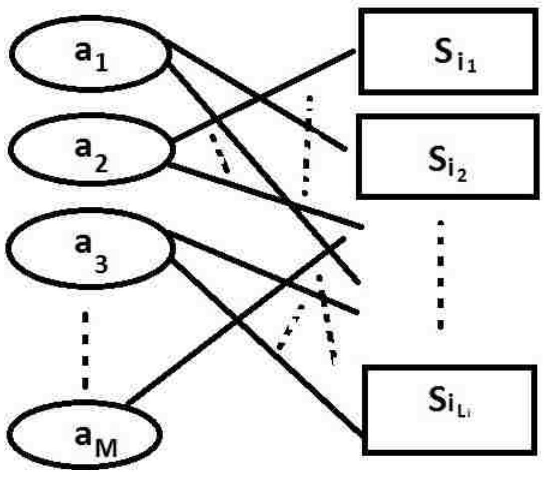

FIGURE 2.13. Factor Graph for a General Case of subgraph G_i

A factor graph consists of two types of nodes; variable nodes each of whom represent one specific variable and factor nodes which represent a function over a subset of variable nodes. My system in question has been demonstrated in the form of an intersection graph in Fig. 2.13.

In my case, the edges from any variable node to a factor node represent a relationship of "$\in$" and the variables connected to the same factor node share a "$\cap$" relationship.

I see that for an upper bound I simply need to remove one (or more) factor nodes from the factor graph; by this method, I am removing some possible combinations which may result in the system finding a desired shape and subsequently stopping. It will then follow that the total average stopping time will become larger for the subgraphs concerned with the removed factor node. Using the same observations, in order to find a lower bound for my system's average stopping time I aim to lessen the requirements of (at least one of) the contributing factors to the stopping time. Such a goal could simply be achieved by removing a number of "$\in$" edges between the variable and factor nodes.

Thus, I will have a number of possible methods of "edge removal" which result in my desirable bounds. I then run my algorithm of finding the average stopping time on trees for these possible methods and choose the one which offers us the tightest average stopping time.

In Figure 2.9 for subgraph G_8, the easiest way of reaching a lower bound would be to remove $\{\mathbf{a_6}\}$ from $\{\mathbf{a_1}, \mathbf{a_5}, \mathbf{a_6}\}$ and $\{\mathbf{a_5}\}$ from $\{\mathbf{a_3}, \mathbf{a_5}, \mathbf{a_6}\}$ and an upper bound by removing one of the factor nodes least likely to occur for every possible situation which results in only $\{\mathbf{a_3}, \mathbf{a_5}, \mathbf{a_6}\}$ and $\{\mathbf{a_5}, \mathbf{a_6}, \mathbf{a_7}\}$ remaining.

2.7 Simulation and Numerical Results

The results of my work are presented in Figures 2.14, 2.15, 2.16 and 2.17 given that the codewords follow a normalized bernoulli distribution with p as the probability of a 1 bit in the codeword as previously explained in Section 2.4.2. As can be seen, the decision factors in Figure 2.10 did not contain any loops and thus no upper or lower bounds were required; instead I directly offer the numerical and simulation results in Figure 2.15. Furthermore, corresponding rates of privacy invasion as calculated through Eq.(2.5) are presented in Figures 2.16 and 2.17 respectively.

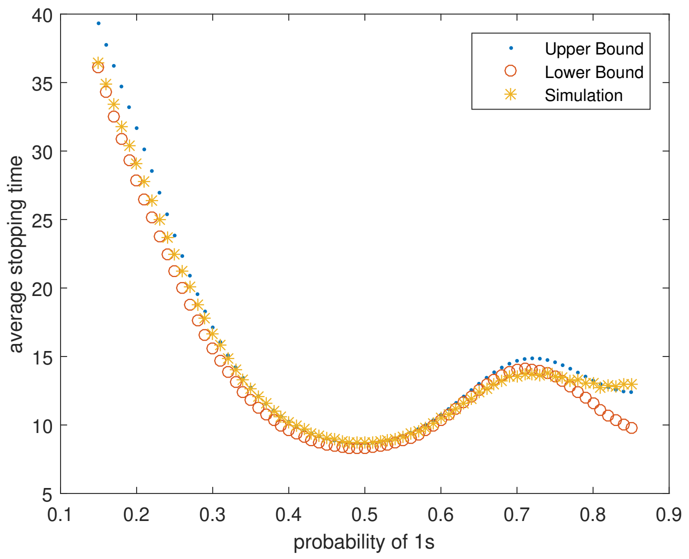

FIGURE 2.14. Numerical and Simulation Results for Subgraphs in Figure 2.9

As can be witnessed in Figures 2.14 and 2.15, average stopping time versus query probability distribution is minimal around the point $p = 0.5$. This is quite understandable seeing as how for any higher or lower amounts of p codeword probabilities incline towards a specific query. This thus result in a higher average stopping time since the frequency of certain queries and the detection of their corresponding decision factors might become more sparse. This conclusion is further based upon the initial inspection that all active subgraphs are equally likely.

However, breach of privacy is not following a specific behavior. In Figure 2.16, a behavior close to that of Figures 2.14 and 2.15 is maintained, while in Figure 2.17, a completely different behavior is witnessed. This contrast in behavior could be traced back to my issues of feasibility as manifested in Section 2.4 where after creating a pseudo table for both Figures 2.5 and 2.6, I witnessed how additional codewords could cause breach of privacy within the original subgraphs. Furthermore, I assumed I was able to remove a number of codewords to make the calculation of decision factors simpler. Thus, the reason why I observe such different behaviors, lays in the dependence between my measure of privacy and the pseudo tables and the number of subgraphs (M) as further suggested by Eq.(2.5).

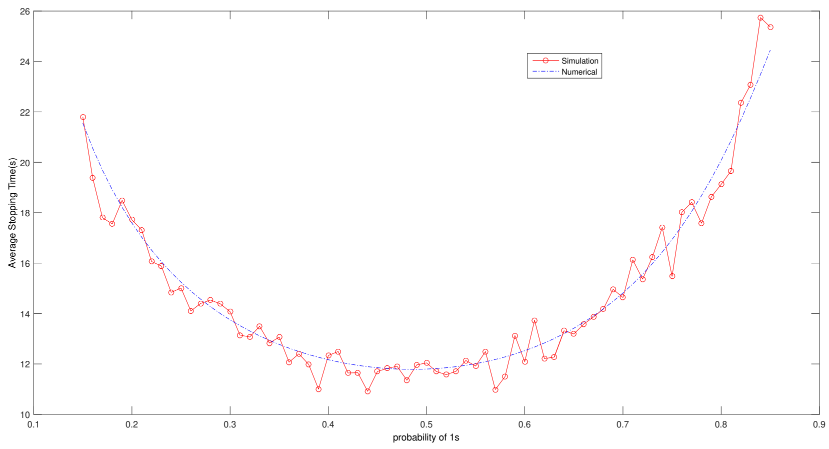

FIGURE 2.15. Numerical and Simulation Results for Subgraphs in Figure 2.10

I started the dicussion with a problem of detecting one of $G_i, 1 \leq i \leq M$ edge-wise disjoint subgraphs in a binary codeword query setting. Every following step represented the necessary issues to accommodate such a problem. I first introduced the concept of feasibility of the problem. I then witnessed how in order to address this issue, I would need to impose minor changes to the original set of subgraphs. These changes then resulted in a breach of privacy within the original subgraphs since I was forced to allow queries concerning separate edges within a certain subgraph to be addressed. Thus the average breach of privacy introduced in the discussion and Eq. (2.5) is specific to the set of original subgraphs.

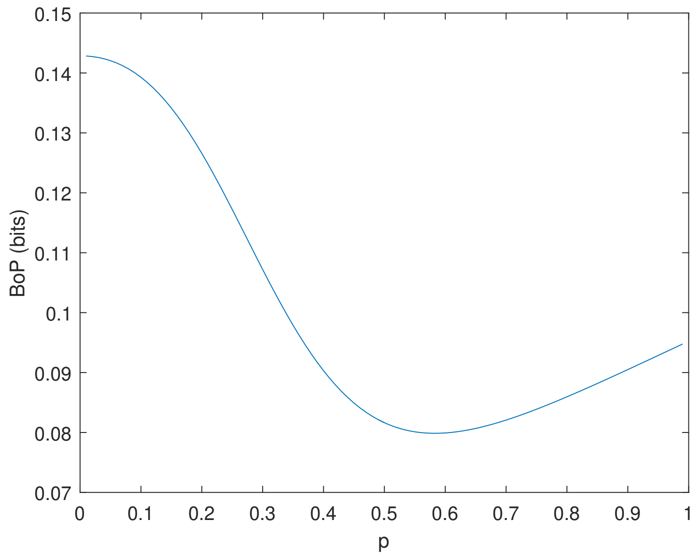

FIGURE 2.16. Breach of Privacy for graph detection in Figure 2.5

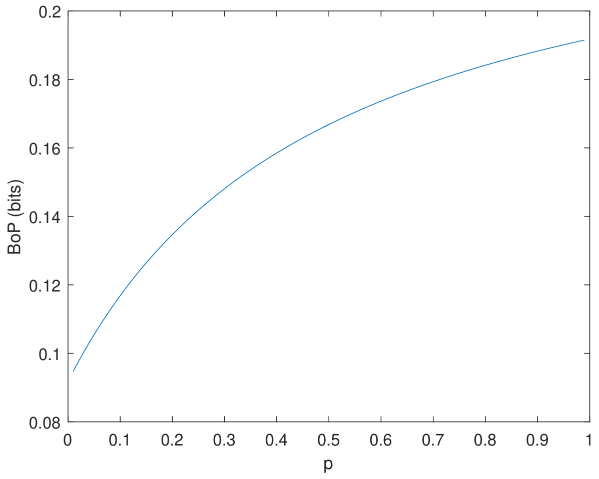

FIGURE 2.17. Breach of Privacy for graph detection in Figure 2.6

It is interesting to note how even the concept of which codewords are chosen could factor in such results. In my work, due to the complexities of calculating decision factors while keeping all 2^N possible codewords, I chose to limit myself to a very limited number of codewords. However, if I had kept all 2^N queries, I would have witnessed more consistent results in the form of breaches of privacy.

However, this realization can help us utilize results such as those depicted in Fig. 16 and 18 as privacy representations of such detection problems (a characteristic of any network of graphs I hope to detect and my specific choice of codewords). I can then use this information about BoP in three manners:

(a) I can assume a threshold of breach of privacy I_{th} I would like to insure and thus choose the range of $p \in [0, 1]$ which holds $BoP(p) \leq I_{th}$. I will then choose the best p within the new range which will help minimize the average stopping time.

(b) Taking into account the relationship between codewords, decision factors and the resulting average stopping times, it follows that aveage stopping time is also a function of the set of codewords used in my process of detection. I can thus assume an overall utility function in the form of the summation of average stopping time and the BoP using a Lagrange coefficient. I then aim to minimize this utility function as a whole (since I aim to minimize both the average stopping time and the leaked information). I could do so by two methods: (1) finding the corresponding $p \in [0, 1]$ which helps satisfy this condition given a specific set of codewords are chosen (as explained in Section. IV-A) or (2) defining a set selection problem over the original $|C|$ codewords for any subgraphs such as those depicted in Fig. 2.5 and 2.6 respectively to find optimal selections which could offer us the best utility function.

2.7.1 Calculation Complexities

Using the steps explained in Algorithm 4, I will simply need to calculate two CDF complements between any two factor nodes j and j' in a factor graph each representing $(1-CDF(S_{i_{l_j}} - S_{i_{l'_j=l_j-1}}))$ and $(1-CDF(S_{i_{l'_j}} - S_{i_{l_j=l'_j+1}}))$. These CDF complements will help us go through every decision tree possible (whichever node might have caused stopping) and fully support calculation transmissions from one factor node to another. Thus if there are V' factor nodes in a certain factor graph I need $2(V'-1)$ CDF complements to fully support step 8 of Algorithm 4. Second, I need to acquire the PMF of V' nodes to calculate the 7^{th} step in the Algorithm 4. This brings us the final conclusion that in a factor graph of a tree nature with V' factor nodes, I require a total of $3V'-2$ calculations to fully recover the average stopping time.

I note that in Eq. (2.7) I showed that the maximum number of vertices in all decision factors will be in order of $O(M|C|)$ assuming $|C| >> M$. It thus follows that the complexity of the final solution will also be in order $O(M|C|)$. It is important to note that the total number of queries $|C|$ is a variable that may be given to us or I may need to find for myself. In the latter case, $|C|$ might be as high as 2^N. Then the overall complexity of the problem will be exponentially large. However, if $|C|$ is a given set of codewords, the complexity levels are much lower.

2.8 Comparison with Generalized Binary Search

It is important to note that given a table of queries and outputs like those offered in Tables 2.1 and 2.2, there are other methods of detection or rather classification as presented in [31, 32]. These papers introduced Generalized Binary Search (GBS) as the fastest method of classification. GBS aims to choose the query which divides the set of possible hypotheses into the two most symmetric sets repeatedly. This means that if I have a total of M possible candidate subgraphs, at the first step of querying, each query will decompose the set of hypotheses into two sets; those with an output of 1 to query $\mathbf{a_j}$ or those with an output 0 to the same query. I then aim to choose the query which offers us the two such sets closest to one another in size. In the next step the two queries whose output division to the two new sets result in the most symmetric sets are chosen and so forth. However, the nature of my query and output tables is different.

In all previous cases of such tables, the outputs were definite, meaning if a subgraph was detected, only this subgraph would be active and all outputs to any query would be consistent. In my work (and both examples) I have cases where the original set of subgraphs could not be detected directly given the nature of my allowed queries. I then opted to decompose my set of subgraphs to allow consistency with the nature of my queries and then attempted to detect such subgraphs. However, such a decomposition would mean that a deterministic decision tree could only detect one and only one decomposed subgraphs rather than the original subgraphs. For example, in Table 2.4 each of the subgraphs G_{411}, G_{412} and G_{42} represent an edge-wise disjoint substructure of the original subgraph G_4. I aim to detect the activity of G_4, which means that I want to assume at any time slot any of the edges within G_4 which may belong to either G_{411}, G_{412} or G_{42} could be active. However, by making a deterministic query (as is done in [31] and [32]) and viewing the result, I move down the decision tree in the direction of detecting one and only one of G_{411}, G_{412} or G_{42}. Now if in the next time slot another one of the 3 possible subgraphs contains the active edge, I may not be able to move down the decision tree since my previous decision and move was based upon another subgraph's activity. Thus, I may get stuck at a certain point of the decision tree and not be able to make any decision about the active subgraph.

On the other hand, in my framework, I assume any of the subgraphs may be active and then simply keep on querying the system until a pattern corresponding to one of them appears. There may be a difference in the active subgraph identity in my sequential queries however, I am not stopping until one certain pattern representing a specific substructure G_{411}, G_{412} or G_{42} appears.

In other words, in my method, I respect the random nature of the table I have created to satisfy my needs by making randomized queries while all other methods of detection are based upon a deterministic table and thus creating a deterministic decision tree. However, a deterministic method cannot satisfy the conditions of a random table. I thus choose to refer to my modified tables such as Tables 2.3 and 2.4 as pseudo tables to represent their randomized nature.

Still, I could assume that the random outputs to different queries are the result of a noisy channel so as to utilize the research done over GBS for noisy cases [17]. It is important to note that I do not have channel noise in this discussion. However, random response to a given query due to random edge activities could be treated as an equivalent noisy channel effect. Unfortunately, any solution offered by [17] needs to introduce an ϵ error range which makes any comparison between my results and those of a GBS nature irrelevant. Furthermore, this revised version of GBS under channel uncertainty does not take advantage of latent structures as a result of decomposition that my approach enables.

Chapter 3

Partition of Random Items: Tradeoff between Binning Utility and Meta Information Leakage

3.1 Introduction

Today internet has become so intertwined with our everyday activities that it is impossible to imagine living without it. However, this dependency could be subject to exploitation by the eavesdroppers. Assuming every browse as a query, it could be argued that the search history of a user contains a series of queries specific to him which could point to his specific likes and dislikes. By following every user's history of browses vital information specific to each user could be developed. It then becomes important that each user attempt to add some level of protection to their browses to hide necessary -or rather enough information about themselves.

One method is the use of proxy websites. Such websites offer the user a URL box where he can input any website he wishes to visit. The only difference is that in such websites, the address input is encoded into a series of characters which appear at the end of the URL of the original proxy website. These characters change through time by seconds meaning if the service provider records the URL opened through the proxy website and decides to open it to access specific content by inputting the URL, he will not go to the encoded web page. Also, such websites slow the connection. However, through this method, the user has the option of choosing multiple proxy websites and thus presumably cutting down on the utility loss. Thus if a utility function based upon connection speed -bandwidth- for the user is calculable, a privacy constrained problem between the user and an eavesdropper (for example a service provider) could be defined. The solution to such a problem could offer insights in regards to the tradeoff between proxy allocation utility and meta information leakage when I face the problems of partitioning a set of random items (i.e. websites that a user has chosen to visit following his own distributions) into a given number of bins (i.e. a given set of proxy servers each of which has its own utility function, as will be further detailed in Section 3.2).

In my proposed framework as detailed in Section 3.2, meta data information refers to the patterns about a sequence of items(for example the user's favorable websites) infer-able based upon a sequence of bins (e.g. proxy sites) observed by an eavesdropper. This assumption is an expansion of [38] and [39]. However, it should be noted that due to usage of proxy sites, an eavesdropper cannot directly observe the original input items, but rather bin indexes. Under my proposed novel framework, I introduce multi-submodularity and submodularity as two means of reducing the complexity level of such problems, namely, dividing M random items into N bins, under an upper-bound on leaked meta information.

This chapter represents an expansion of my previous works in [5, 6] where in [5] a utility function was introduced in the form of the average stopping time for detection of an active subgraph using certain queries while in [6], I developed a concept of information leakage through a vast set of possible queries. In this chapter, I shift my attention from detection of an active subgraph to seeking tradeoff between optimizing utility of partitioning a set of random items and restraining information leakage.

The concept of privacy has already been explored in many works such as [40, 41] where a general but non-mathematical explanation was offered. However; in my work, I go into further details as to what privacy represents in my framework and how it could be formulated into many settings. Later in this chapter, I find it necessary to utilize the concept of multi-submodular set function problems and their solutions. This concept was widely discussed in [42] where they introduced a series of sufficient conditions on multi-submodular set functions by which the multi-submodular problem could be transformed into a submodular set function problem. Then, further discussions about the existence of a solution to the new problem were made. By doing so -and if a solution was proven to exist-, the complexity of the problem could be shown to be reduced from NP to polynomial. However, [42] did not offer any algorithmic solutions in such cases.

The novel contributions of this chapter are as follows: (1) I introduce a new multi-agent multi-variant optimization problem with a privacy leakage constraint offering specific accommodations for online browsing which turns out to be NP-complicated; (2) I introduce the novel idea concerning dual nature of the multi-agent optimization problems and their corresponding implications in such problems; (3) I utilize multi-submodularity property to prove the existence of a transformation to submodular set problems given a series of sufficient conditions and then concluded the existence of a polynomial solution; by doing so, I simplify the complexity of the problem from NP to polynomial (4) I offer a new less restrictive set of sufficient conditions as well as a submodular set function optimization solution algorithm with its corresponding calculation complexities for the specific case of $N = 2$.

The rest of the chapter is organized as follows. In Section 3.2 I formulate the problem in terms of privacy and utility functions. I dissect what the goal and the constraints are. In Section 3.3, I first introduce a revised version of the utility function and then find the sufficient conditions under which this function is equipped with multi-submodular property. Due to general limited knowledge about multi-submodular solutions and how they are developed, I then assume a specific case $N = 2$. In Section 3.4 an algorithm for this specific case is developed which has a polynomial cost and an accuracy of 0.432. Finally, in Section 3.6, I conclude the discussion by reiterating what I have accomplished in this chapter.

3.2 System Model

First, I propose an abstract framework to formalize the goal of seeking partition of N items into M bins. More specifically, I aim to allocate each one of $1 \leq i \leq M$ possible items (queries) to one of N output bins. There could be at most N^M such partitions. It follows that any set allocation $A_l, 1 \leq l \leq N^M$ results in N sets $S_j^{(l)} \subseteq \{1, 2, ..., M\}, j = 1, 2, ..., N$. Each such set is defined as

$$S_j^{(l)} = \{i | \theta_{i,j} = 1\}$$
$$where \quad \theta_{ij}^{(l)} = \begin{cases} 1 & i \in S_j^{(l)} \\ 0 & i \notin S_j^{(l)} \end{cases} \tag{3.1}$$

I further assume $S_j^{(l)} \cap S_k^{(l)} = \emptyset, j \neq k$. Furthermore I have $\bigcup_{j=1}^{N} S_j^{(l)} = \{1, 2, ..., M\}$. Finally the size of each such set $S_j^{(l)}$ is defined as $L_j^{(l)}$.

3.2.1 Probabilistic Model

I assume at any time slot one and only one of the inputs is chosen with a certain probability. Thus if I use variable $X \in \{1, 2, ..., M\}$ as a representation of set of items, I could have $P(X = i) = P(\gamma_i = 1) = \pi_i, 1 \leq i \leq M$ as a representation of the probability of choosing item i from the set X where $\gamma_i \in \{0, 1\}$. It further follows that $\sum_{i=1}^{M} \gamma_i = 1$, stipulating that one and only one of M items is selected. These M items could represent a set of M web pages to be visited by a user at a particular time instant. The prior probability distribution of M items reflects the user's favoritism toward these web pages. .

Next, I introduce an observable random variable $Y \in \{1, 2, \cdots, N\}$, denoting the index of the bin (the proxy site) employed to carry one of the $M > N$ items. It follows that the probability of each bin's appearance given a set allocation scheme such as A_l will be equal to

$$P(Y=j|A_l) = \sum_{i=1}^{M} P(Y=j|A_l, X=i)P(X=i|A_l)$$
$$= \sum_{i=1}^{M} P(Y=j|A_l, X=i)P(X=i) \tag{3.2}$$

where I have dropped the second conditional probability due to the independence between X and A_l. Furthermore $P(Y = j|A_l, X = i) = \theta_{ij}^{(l)} \in \{0, , 1\}$. It thus follows that

$$P(Y=j|A_l) = \sum_{i=1}^{M} \theta_{ij}^{(l)} P(X=i) \rightarrow$$
$$P(Y=j|A_l) = \sum_{i \in S_j^{(l)}} \pi_i = \alpha_j^{(l)} \tag{3.3}$$

3.2.2 Revealed Information

By choosing to allocate M items to N bins where $N \leq M$, I have injected ambiguity and uncertainty into the output binning index sequence about the input item sequence over a successive n visits or channel uses. In other words, if I originally chose to transmit n of such items, my total set of possible sequences would be of form $\overrightarrow{\mathbf{X^n}} = [X_1X_2...X_n]$ out of M^n possible outcomes. From an observer's perspective which can only have access to which one of N bins is deployed in each time slot, sequences in the form of $\overrightarrow{\mathbf{Y^n}} = [Y_1Y_2...Y_n]$ has cardinality of at most $N^n < M^n$. Despite the amount of uncertainty added due to the many-to-one mapping between items and bins, the output sequence sill reveals certain amount of information regarding the patterns of sequences of M random items.

This observation could be further studied by indicating how my allocation system resembles a coding framework where I have an equivalent channel whose input variable is X and output Y, as show in Figure 3.1.

Under such a framework, the equivalent channel output sequence $\overline{\mathbf{Y}}^n$ can help an eavesdropper classify the input sequence $\overline{\mathbf{X}}^n$ into a number of differential classes.

As a result, information about the specific input item patterns is leaked to certain degree and can be measured using conditional mutual information $I(X;Y|A_l)$ between X and Y, given a particular channel mapping (i.e. partition A_l relationship as illustrated in Figure 3.1.

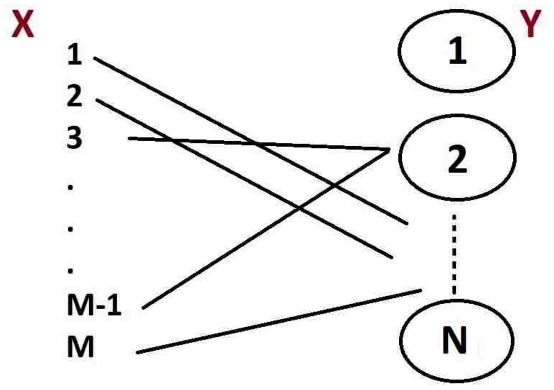

FIGURE 3.1. Coding Channel Representation of the Problem

Such conditional mutual information thus measures the maximum number of bits of meta information about item sequence per channel use. Therefore, I can have at most $2^{nI(X;Y|A_l)}$ sequences $\overline{\mathbf{X}^n}$ distinguishable by inferring based on $\overline{\mathbf{Y}^n}$. I thus adopt $I(X;Y|A_l)$ as the privacy metric conditioned on a particular partition mapping A_l.

It follows that due to the combinatorial nature of a set allocation problem there are a total of N^M possible methods to allocate these M items to the N sets. I can formulate the mutual information over a set allocation $A_l, 1 \leq l \leq N^M$ as:

$$I(X;Y|A_l) = H(X|A_l) - H(X|Y, A_l) =$$
$$H(Y|A_l) - H(Y|X, A_l) = H(Y|A_l) =$$
$$H(\alpha_1^{(l)}, \alpha_2^{(l)}, ..., \alpha_N^{(l)}) \tag{3.4}$$

where I have used the notion of $H(a_1, a_2, ..., a_m) = -\sum_{v=1}^{m} a_v \log a_v$ and the fact that $H(Y|X, A_l) = 0$ because if both the input X and the channel scheme A_l are known, then output Y could simply be calculated.

3.2.3 Utility Function

Note that the main reason why I chose bin allocation was to reach a higher utility function. In this section I define a utility function to apply to the problem. If an allocation scheme A_l has resulted in $S_1^{(l)}, S_2^{(l)}, ..., S_N^{(l)}$, I assume each of the N bins offer a utility function of their own based upon the set they have been bestowed. Every bin j thus offers a utility represented by $f_j(S_j^{(l)})$. It is important to note that f_j represents a set function meaning it would change as different subsets of the universal set are chosen. An example for such a function would be if $f_j(S_j^{(l)}) = f_j(|S_j^{(l)}|)$ meaning the function changes as the number of members within the set $S_j^{(l)}$ changes.

It then follows that the average utility function will be in the form of $U_l = \sum_{j=1}^{N} f_j(S_j^{(l)})P(Y = j|A_l) = \sum_{j=1}^{N} \alpha_j^{(l)} f_j(S_j^{(l)})$.

3.2.4 Problem Definition

Based on the previous observations I set a goal to find the set allocation A_l over which (a) U_l is maximized and (b) $I(X; Y|A_l) \leq I_{th}$ where I_{th} represents the maximal allowed revealed information.

I aim to gather both the utility and the constraint imposed in the form of one function I hope to maximize. Thus, a new maximization problem could be developed:

$$\max_{1 \leq l \leq N^M} \quad U_l + \lambda(I_{th} - H(\alpha_1^{(l)}, \alpha_2^{(l)}, ..., \alpha_N^{(l)})) \tag{3.5}$$

where $\lambda \geq 0$ represents a variable connecting the utility and the constraint to each other so as to allow comparison between them. I can further express this equation by opening it as:

$$\max_{1 \leq l \leq N^M} \quad \sum_{j=1}^{N} [\alpha_j^{(l)} f_j(S_j^{(l)}) + \lambda \alpha_j^{(l)} \log(\alpha_j^{(l)})] \tag{3.6}$$

I can now see that if I define $F_j^{(l)} = \alpha_j^{(l)} f_j(S_j^{(l)}) + \lambda \alpha_j^{(l)} \log(\alpha_j^{(l)})$, Equation (3.5) is simply a sum of functions defined over a series of sets. I refer to these as multivariate set functions seeing as how their values are based upon specific sets and variables introduced in each of these sets.

3.3 Multi-Submodular Set Functions as a Means of Solution

As mentioned previously, the problem formulated in Eq. (3.5) is NP-complicated (it is solved when a search over N^M possible set allocations is carried out and the best allocation is chosen). Still, I could opt to utilize the definition of multi-submodular set functions so as to reduce the complexity to that of polynomial at the cost of accuracy. In order to do so, I raise concerns about possible solutions. Next, I offer insight as to how I could deal with each case.

3.3.1 Imposing Multi-submodularity

In [42], it was shown that if I can prove multi-submodularity for functions such as those formulated in Eq. (3.5), then they could be modeled as simpler problems (submodular set functions). I thus, aim to find the sufficient conditions for such occurrence. To do so, I first offer a review of multi-submodularity.

As mentioned in [42], if I define $\mathbb{M} = \{1, 2, ..., M\}$, then a multivariate function $F : (2^{\mathbb{M}})^N \to \mathbb{R}^+$ is multi-submodular if for all pairs of tuples $(S_1, ..., S_N)$ and $(T_1, ...T_N) \in (2^{\mathbb{M}})^N$ I will have:

$$\begin{aligned} F(S_1, ..., S_N) + F(T_1, ..., T_N) \geq F(S_1 \cup T_1, ..., S_N \cup T_N) \\ + F(S_1 \cap T_1, ..., S_N \cap T_N) \end{aligned} \tag{3.7}$$

Since in my formulation functions are separately defined on different sets, the condition in Eq. (3.7) is simplified to the sufficient condition of submodularity of $F_j^{(l)}$ for all sets S_j for Equation (3.5). I now need to find the sufficient condition for submodularity of $F_j^{(l)}$ when defined over a set S_j.

3.3.2 Imposing Separate Submodularities

For an easier mathematical representation of the following derivation I denote $F(S_j) = F_j^{(l)}$. Furthermore, I denote $f_j(S_j^{(l)}) = f(S_j^{(l)})$. Both these denotations allude to the fact that once a set allocation A_l is chosen, its index could be dropped.

In the next step, I opt to use diminishing return property as the means of making certain each of these functions are submodular. Following is a definition of diminishing returns for submodular functions, after which I derive the sufficient conditions for the case discussed in Eq. (3.5).

Diminishing Property Return dictates that if I define $\mathbb{S}$ as the universal set, a set function $F : 2^{\mathbb{S}} \rightarrow \mathbb{R}^+$ is submodular if, for all $A, B \subseteq \mathbb{S}$ with $A \subseteq B$ and for each $x \in \mathbb{S} - B$ I have [51]:

$$F(A \cup \{x\}) - F(A) \geq F(B \cup \{x\}) - F(B) \tag{3.8}$$

Now I attempt to expand Eq. (3.8) for each $F(S_j)$. However, to properly do so, I first need to account for the behavior of this function.

I have defined $F(S_j) = \alpha_j f(S_j) + \lambda \alpha_j \log(\alpha_j)$ where it seems that the function has a singular relationship with set S_j. However; there is a secondary relationship the function shares with the set $S_j^{\mathbf{C}} = S - S_j$ where $S = \mathbb{S}$ represents the universal set. This relationship could be modeled as $F(S_j^{\mathbf{C}}) = (1-\beta_j) f(S - S_j^{\mathbf{C}}) + \lambda(1 - \beta_j) \log(1 - \beta_j)$ where I have used the fact that $\beta_j = 1 - \alpha_j$ seeing as how I define

$$\beta_j^{(l)} = \sum_{i \in \{S - S_j^{(l)}\}} \pi_i \tag{3.9}$$

Thus, for any set S_j, I must find the sufficient conditions for the existence of diminishing property for both functions $F_1(S_j) = \alpha_j f(S_j) + \lambda \alpha_j \log(\alpha_j)$ and $F_2(S_j) = (1 - \alpha_j) f(S - S_j) + \lambda(1 - \alpha_j) \log(1 - \alpha_j)$. To do so, I will evaluate their necessary conditions and then find their intersection as the final conditions (assuming they do not negate one another).

Note: For any further references, I first need to address a series of variable and function definitions which are going to play a vital role in the rest of this chapter:

Definitions

1. Any variable represented with a capital Letter represents a set.
2. Any variable represented with a small letter represents an element.
3. $A - B$ represents a set containing all elements of set A which do not appear in set B.
4. α_x represents the probability of item x and α_A represents the sum of probabilities of items mapped into a set A.
5. α_{BA} represents the difference in the sum of probabilities of items mapped into the sets B and A which could be further shown as $\alpha_{BA} = \alpha_B - \alpha_A$.
6. $g(C, D)$ represents the 1^{st} order difference of a set function $f(C)$ from $f(C-D)$ where $D \subseteq C$ which could be formulated as $g(C, D) = f(C) - f(C - D)$.
7. $q(C, C_1, D, D_1)$ represents the 2^{nd} order difference of a set function $f(C)$ where $C_1 \subseteq C$ and $D_1 \subseteq D$ which could be formulated as $q(C, C_1, D, D_1) = g(C, D) - g(C_1, D_1)$.
8. I assume the probability of items is sorted in a decreasing manner such as $\pi_1 \geq \pi_2 \geq ... \geq \pi_M$.

Theorem 3.3.1. *The set functions $F_1(S_j)$ and $F_2(S_j)$ and as a result $F(S_j)$ are submodular if*

(1) $g(S_j, S_w) \leq 0$

(2) $q(S_j, S_u, S_w, S_y) \leq 0$

(3) $|g(S_j, S_w)| \geq \lambda \log{(\frac{1}{\pi_M})}$

for all possible sets $S_w \subseteq S_j \subseteq S$ and $S_u \subseteq S_j$ and $S_y \subseteq S_w$ where S is the universal set.

The proof for this lemma is presented in the Appendix under Theorem III-1. In the proof, a series of sufficient conditions for either $F_1(S_j)$ and $F_2(S_j)$ are evaluated separately. This is done because although their conditions turn out to be the same, their derivations are vastly different as require separate discussions. It then follows that since both functions require the same set of sufficient conditions, the function $F(S_j)$ which represents either of them being chosen, also follows the same set of sufficient conditions. In the proof, I rewrite inequality (3.8) for set function $F_1(S_j)$, start factorizing α_A, α_{BA} separately and α_x and 1 together and impose sufficient conditions so that each of their coefficients is always positive.

Unfortunately, [42] does not provide us with an algorithm to remodel my multi-submodular problem in a submodular problem, they simply prove that this could be done. Thus, in order to expand upon the idea of polynomial complexity of solution algorithms I opt to assume $N = 2$ and offer the reader the algorithm to deal with such a specific case. I then calculate the complexity imposed by the algorithm to further stress the benefits of using such an idea in spite of accepting error.

3.3.3 Specific Case of $N = 2$

As mentioned previously, in order to show the applicability of submodular functions I choose to reiterate the utility function dictated in Eq. (3.5) for when $N = 2$:

$$\begin{aligned} \max_{1 \le l \le 2^M} \quad & \alpha_1^{(l)} f(S_1^{(l)}) + (1 - \alpha_1^{(l)}) f(S - S_1^{(l)}) \\ & + \lambda(I_{th} + \alpha_1^{(l)} \log(\alpha_1^{(l)}) + (1 - \alpha_1^{(l)}) \log(1 - \alpha_1^{(l)})) = T(S_1) \end{aligned} \tag{3.10}$$

As can be seen, the problem is still exponentially complex seeing as how I need to search over 2^M possible solutions to find the optimal. Thus, once again I aim to impose multi-submodularity (in this case simplified to submodularity) on the new utility function. For the utility function above the same results derived for a general N could be used as a set of sufficient conditions. However, taking into account the joint relationship between the 2 sets and writing the same Inequality (3.8) for Equality (3.10) I am able to find a less restrictive set of sufficient conditions for the submodularity of this utility function as indicated below:

Lemma 3.3.2. *When $N = 2$, the function in Eq. (3.10) is submodular if*

(1) $g(S_j^{(l)}, S_w^{(l)}) \leq 0$

(2) $2|g(S_j^{(l)}, S_w^{(l)})| \geq \lambda \log(K), K < (\frac{1}{\pi_M})^2$

(3) $q(S_j^{(l)}, S_u^{(l)}, S_w^{(l)}, S_y^{(l)}) \leq 0$

for all possible sets $S_w^{(l)} \subseteq S_j^{(l)} \subseteq S$ *and* $S_u^{(l)} \subseteq S_j^{(l)}$ *and* $S_y^{(l)} \subseteq S_w^{(l)}$ *where* S *is the universal set.*

The proof for this lemma is presented in Appendix under Lemma III-2. In the proof, I rewrite inequality (3.8) for set function described in Eq. (3.10), start factorizing α_A, α_{BA} separately and α_x and 1 together and impose sufficient conditions so that each of their coefficients is always positive. In the following section, I will offer a method of solving a problem as introduced in Eq. (3.10).

3.4 Submodular Solution

Starting by [52] there has been monumental work done over greedy algorithms with constraints (as long as they introduce down-monotone solvable polytopes) with a solution proximity of $\frac{1}{e}$. Later [53] introduced a solution proximity of 0.372. Finally [54] proved that this proximity could be increased to 0.432 in maximization problems which is quite close to the no-constraint solution of a symmetric problem.

It is important to note that while all these papers dealt with the issue of constraints, they assumed much more complex constraints than I am dealing with in this discussion. My only constraint is that $\alpha_1^{(l)} \leq \alpha_s$ where I assume $h(\alpha_s) = I_{th}, \alpha_s \leq 0.5$ which is obviously a down-monotone constraint. Thus, I can simply use the results from their work to create my own algorithm to find the submodular function solution to my problem. I present:

Algorithm 1: Submodular Function Solution to the problem as described in Eq.(3.5)

```
1. Let $S_1 = argmax_{e \in X=\{1,\dots,M\}} T[S_1 = \{e\}]$ while $|0.5 - \alpha_1^{(l,1)}| \geq |0.5 - \alpha_s|$.
```

```
2. If there is an element $e \in X \setminus S_1$ such that $T[S_1 + \{e\}] \geq T[S_1]$ and $|0.5 - \alpha_1^{(l)} - \alpha_e| \geq |0.5 - \alpha_s|$, let $S_1 = S_1 + \{e\}$.
3. If there is an element $e \in S_1$ such that $T[S_1 \setminus \{e\}] \geq T[S_1]$ and $|0.5 - \alpha_1^{(l)} + \alpha_e| \geq |0.5 - \alpha_s|$, let $S_1 = S_1 - \{e\}$. Go to Step 2.
4. Return maximum of $T[S_1]$ and $T[X \setminus S_1]$.
```

where I know that at the very last step $T[S_1] = T[X \setminus S_1]$. Now I opt to calculate the complexities of this method. Steps 2 and 3 could repeat $(M-1)+(M-2)+...+1 = \frac{M(M+1)}{2}$ times each while every item could be removed and thus replaced a total of $2M$ times. Thus the total complexity of steps 2 and 3 is equal to $M^2(M+1) = O(M^3)$. The complexity of step 1 is also equal to M. Thus the total complexity of the solution is equal to $O(M^3)$.

This polynomial solution simply makes certain the maximal function obtained is at least 0.432 times the optimal objective function. This range of error occurs because in this method, I am removing and adding members from and to the set S_1 one by one. Thus, at each decision point I am making one locally optimal decision. However, it is widely known that a locally greedy method is not necessarily globally optimal [55].

3.5 Examples and Comparison

In this section I aim to offer the reader two problems where I am hoping to use the results gathered in Lemma 4.3.2 to first find a proper utility function given the specifics of each case. I will then follow Algorithm 1 and compare its results with that of an exhaustive search to compare the two methods in terms of complexity and exactness of the solution in both problems. In both examples, I assume $M = 4, N = 2, \lambda = 0.25, I_{th} = 0.4$ and that $f(S) = f(|S|)$ where S and $|S|$ represent any set and its cardinality respectively. The only difference between the two examples would then lay in their item probability distributions.

3.5.1 Utility function for $\pi_1 = \pi_2 = \pi_3 = \pi_4 = 0.25$

One of the simplest utility functions which could satisfy the conditions presented in Lemma 4.3.2 is a quadratic function in the form of $f_1(|S|) = a|S|^2 + b|S| + C$. By calculating the 1st and 2nd order derivatives I can see that they follow the form of $g_1(|S|) = 2a|S| + b - a$ and $q_1(|S|) = 2a$ respectively. I could then have:

$$2a \leq 0 \rightarrow a \leq 0$$
$$2a|S| + b + a \leq -\frac{\lambda \log (K)}{2}, |S| = 0, 1, ..., 4 \tag{3.11}$$

where by using the worst case scenario I attempt to find the tightest upper bound for b between any case of $|S|$. I will then have:

$$b \leq \frac{-\lambda \log (K)}{2} + a \tag{3.12}$$

Finally, I must have

$$f_1(|S|) \geq 0 \rightarrow c \geq -a|S|^2 - b|S|, |S| = 0, 1, ..., 4 \tag{3.13}$$

Since $K < (\frac{1}{\pi_M})^2 = 16$, I assume $K = 8$ and by further assuming that $a = -1$, I can find acceptable amounts for both $b = -2$ and $c = 25$. Thus, the overall utility function will be in the form of $f_1(|S|) = -|S|^2 - 2|S| + 25$.

3.5.2 Utility function for $\pi_1 = 0.5, \pi_2 = 0.25, \pi_3 = \pi_4 = 0.125$

Once again I assume a quadratic function in the form of $f_2(|S|) = a|S|^2 + b|S| + C$. By calculating the 1st and 2nd order derivatives I can see that they follow the form of $g_2(|S|) = 2a|S| + b - a$ and $q_2(|S|) = 2a$ respectively. The mathematics will be the same as depicted in Eq. (3.11) and (3.12) and (3.13). Since $K < (\frac{1}{\pi_M})^2 = 64$, I assume $K = 32$ and by further assuming that $a = -1$, I can find acceptable amounts for both $b = -2$ and $c = 25$. Thus, the overall utility function will be in the form of $f(|S|) = -|S|^2 - 2|S| + 25$. It is important to note that while the two utility functions for two different sets of item probability distributions are the same, the results of my methods might be different as I will see in Subsection 3.5.3.

3.5.3 Solution Comparison

The results of running Algorithm 1 on the two scenarios have been gathered in Table 3.1. Each cell represents the maximum overall utility achieved in either case by each method where it is obvious that the probability distribution plays a major role on the exactness of the solution compared to the utility function - which is the same in both cases. Also of interest is the negligible loss of utility at $1-\frac{16.8614}{16.8972}=0.0021$ at a desirable cost reduction from NP to P which helps justify my persistence on utilizing submodular solutions.

TABLE 3.1. Solution Exactness Comparison

scenario number	exhaustive search solution	worst submodular set solution
1	16.85	16.85
2	16.8972	16.8614

3.6 Conclusions

In this chapter, I introduced and formulated a problem widely regarded in online browses. To do so, I revisited the concept of privacy leakage discussed in both other and my own previous publications. I further introduced a utility function based upon the user's utilization of the network. I showcased how the problem formulation results in a multi-agent multi-variate problem which is NP-complicated. I then introduced the concept of submodularity and multi-submodularity which help reduce the complexity of such problems to that of a polynomial at the cost of some accuracy. I derived a series of sufficient conditions which would guarantee the existence of a solution. To do so, I introduced a novel observation of the behavior of such multi-agent multi-variate set functions which I called duality. Once the existence of such solutions was guaranteed, I introduced algorithms that could help us when $N = 2$ (but the original complexity is still combinatorial) and showcased how they help reduce the complexities.

Chapter 4

Partition of Random Items: Tradeoff between Binning Utility, Meta Information Leakage and Hypotheses Distinguishability

4.1 Introduction

In our daily use of internet there are two very specific issues of privacy. The first issue rises when we aim to hide as much information about ourselves while still enjoying the service of web -which is always prone to eavesdrops. As an example, we hope to use the Google search engine without a third party studying our search history and learning about us. For future references, we refer to this class of problems as Utility versus Privacy (UvP).

The second scenario rises when we wish to be distinguished from other users but not reveal all the information we share on the web. As an example, when we surf Amazon, based upon our previous searches over the website, we are offered new suggestions we may be interested in. However, we hope to keep our privacy intact to some level at the same time. If we search for a specific political ideology, we do not like to have this ideology be revealed to a third party. If we were, we would feel uneasy about how much private information about us has leaked. This class of problems will be referred to as Utility versus Privacy and Distinguishability (UvPD).

It could then be deduced that in both scenarios we aim to maximize a utility reflected by my browses (a good internet service in the first case and suggestion of a relating product in the second) while keeping the privacy reflected by patterns shown in sequence of webpages visited and obscured by the underlying browses to a certain upper bound. It could be argued that the best solution for such invasion of privacy problems for a user is to find approaches seeking tradeoff between some utility and the resulting privacy leakage.

One method to carry this out, is the use of website proxies. Such websites offer the user complete anonymity by coding the IP address of the visited web pages into a series of characters which can no longer be traced. The only detriment to such proxy websites is their limited bandwidth and time-outs.

As a solution, we can employ multiple proxy websites and thus cut down on the utility loss. However, by using multiple proxy sites we are allowing a level of privacy breach into my browses where the eavesdropper is able to deduce some information about my tendencies by observing the distribution of used webpages over multiple networks. Then, if a utility function based upon connection speed -bandwidth- for the user is calculable, a privacy constrained problem between the user and an eavesdropper (for example a service provider) could be defined.

I next elaborate on the relationships between the terms used throughput the chapter: utility, information leakage, and hypothesis distinguishability.

Utility represents the advantage received by using a number of bins. Each of these bins may have their own specific advantages and thus utility represents a sum of such advantages over all bins.

Once an item is mapped into a specific bin, its exact properties are obscured by the properties of the bin meaning it becomes indistinguishable from other items mapped into the bin. This is called **information protection**. On the other hand, a bin reveals a certain level of information about the items within it. This is called **information leakage**. The privacy leakage I quantify is thus via patterns of long sequence of binning items showing up to an observer.

Hypotheses distinguishability represents a measure of difference between multiple modes where based upon the frequency of different outputs it could be deduced that a certain other mode is activated. Thus distinguishability offers an insight into the statistics revealed over long sequence of symbols.

To formalize such trade-off problems, I propose a partitioning framework in which a set of M items (e. g. websites) are placed into N bins (e. g. proxy sites) to maximize a certain payoff function while meeting constraints imposed due to concerns of privacy leakage or distinguishing capability. Then, a given distribution $G_1, G_2, ...$ on the random items represents a particular operating mode or behavior over a larger scale, which the user is willing to be classified. Under a particular distribution, however, a user is more concerned of the sequence patterns, which could be inferred to certain extent by the resulting patterns of bins (e. g. proxy sites) deployed.

my overarching goal is to pose and then solve the underlying constrained combinatorial optimization problems by first considering the tradeoff between meta leakage and binning utility, and then to further add the distinguishability into the objective by seeking optimal binning of items. Due to the exponentially growing cost in searching such partitions, I instead seek sufficient conditions under which polynomial order complexity algorithms exist by resorting to multi-submodular and submodular structures in my problems.

4.1.1 Related Works

The concept of privacy has already been explored in many works such as [40, 41] where a general but non-mathematical explanation was offered. However, in my work, I offer a novel representation of privacy through the introduction of typical sequences.

It is important to note that one of the most prominent measures of privacy as described in works such as [45] has been differential privacy. Differential privacy is mainly concerned with limiting the information leaked through different sequences generated by a randomized process. In my problem settings, differential privacy aims to investigate the effects of a minimal change in the input in the overall output of the system. In other words, if there are originally X^n sequences at my disposal which are mapped to Y^n sequences in the output, is there a minor change in the input sequence that could result in an out of control change in the output sequence?

Overall, differential privacy in literature is concerned of one-shot measure ([46]). However, all my measures are in average sense including average utility, mutual information, KL divergence, whose functional significance rests upon repeated drawing from a distribution over long run. Thus differential privacy is not of concern here. Even if I could consider differential privacy in the settings of long sequences, such privacy measure is not quite relevant due to the long-sequence constraints.

As for the addition of distinguishability, to the best of my knowledge, the closest work to mine was done by [47] where they considered the tradeoff between distinguishability and information leakage when the former is quantified using KL-distance and the latter using mutual information. However, they formulated the problem using the concept of missed detection and false alarm as the basis of hypotheses testing. In this work, the authors were interested in a randomized binning process which tended to be more complex than my routine. Such a goal came at the cost of sacrificing the concept of trade-off by only allowing very small privacy leakage (less than ϵ) in order to invoke a first order approximation to reduce the computational complexity of the optimization problems. However; in my work, I am concerned with any amount of trade-off with deterministic binning to seek suboptimal and polynomial order approximations enabled by the fundamental properties of multi-submodular set functions.

The multi-submodular function was discussed extensively in [42] where they introduced a series of sufficient conditions on multi-submodular set functions by which the multi-submodular problem could be transformed into a submodular problem. Then, further discussions about the existence of a solution to the new problem were made. By doing so -and if a solution were proven to exist-, the complexity of the problem could be shown to be reduced from NP to polynomial. Such sufficient conditions have been adopted by us in in seeking proper binning utility functions under which such sufficient conditions hold true.

4.1.2 Chapter Contributions

It is important to note that some of the problems introduced in this chapter, were partially introduced in my previous work [49] namely the first problem introduced in the chapter (UvP). However much novel required discussion of the material is presented here. Furthermore, in this chapter I introduce a new class of problems namely UvPD which represents a much more complicated scenario. Even more, in this work, I go into full details to address all issues and ideas intertwined with my work. Finally, a more thorough set of numerical results is presented.

More specifically, the major results obtained are itemized below: (1) a measure of distinguishability between the $K = 2$ hypothesis; (2) a measure of average leaked information given any of the K hypothesis is active; (3) a formulation of a multi-agent multi-variant optimization problem with a privacy leakage constraint; (4) the development of insight into the complexity of such an NP-hard problem and further sufficient conditions under which I can simplify it into a polynomial problem; (5) a description of the algorithm utilized to find the solution given the sufficient conditions followed by the proximity results and finally (6) numerical examples to further demonstrate the applicability of submodular solutions, as compared with the results using exhaustive search in manageable settings.

It is important to further discuss the contributions of this chapter in further detail to fully acknowledge the novelty of my results:

Firstly, to the best of my knowledge, a binning representation of utility versus privacy or distinguishability versus either of these two has never been developed before and is thus well-worthy of a more in-depth study.

Secondly, during the proof of sufficient multi-submodularity conditions, I came across the concept of duality in finite size binning problems where simply satisfying the submodularity condition for a series of set functions does not necessarily guarantee the multi-submodularity of the entire multi-set function. In order to deal with this issue, I introduced the concept of imposing submodularity on the complements of different sets. This poses a further set of sufficient conditions on the overall set utility function. Then I discussed how the intersection of sufficient conditions on both the original sets and their complements will result in the required overall sufficient conditions. my in-depth study of the imposed duality in such settings, to the best of my knowledge, has never been done before. In most cases such as [42], it has simply been assumed that the number of items is infinite making sure that a one way satisfaction of submodularity will be enough to guarantee the overall convergence of the methods introduced later on, which unfortunately can not be applied in my case with finite number of items. I thus offer my own spin on the same concept to account for this new problem.

4.1.3 chapter Organization

The rest of this chapter is organized as follows. In Section 4.2 I formulate the problems in terms of privacy and utility functions. I dissect what the goal and the constraints are. In Section 4.3, I inspect the overall utility functions of each problem and then find the sufficient conditions under which they are equipped with multi-submodular structure. In Section 4.4 an algorithm for each case is developed offering a polynomial complexity and an accuracy level of e or $\frac{1}{e}$ depending on whether the goal is to minimize or maximize the overall utility function respectively.

4.2 System Model

First, I propose an abstract framework to formalize the goal of seeking partition of M items into N bins. More specifically, I aim to allocate each one of $1 \leq i \leq M$ possible items (queries) to one of N output bins. There could be at most N^M such partitions. It follows that any set allocation $A_l, 1 \leq l \leq N^M$ results in N sets $S_j^{(l)} \subseteq \{1, 2, ..., M\}, j = 1, 2, ..., N$. Each such set is defined as

$$S_j^{(l)} = \{i | \theta_{i,j}^{(l)} = 1\}$$
$$where \quad \theta_{ij}^{(l)} = \begin{cases} 1 & i \in S_j^{(l)} \\ \\ 0 & i \notin S_j^{(l)} \end{cases} \tag{4.1}$$

I further assume $S_j^{(l)} \cap S_k^{(l)} = \emptyset, j \neq k$. Furthermore I have $\bigcup_{j=1}^N S_j^{(l)} = \{1, 2, ..., M\}$. Finally the size of each such set $S_j^{(l)}$ is defined as $L_j^{(l)}$.

4.2.1 Probabilistic Model

I assume at any time slot one and only one of the inputs is chosen with a certain probability. Thus, assuming hypotheses G_p is active and if I use variable $X \in \{1, 2, ..., M\}$ as a representation of set of items, I could have $P(X = i|G_p) = P(\gamma_{i_p} = 1) = \pi_{i,p}, 1 \leq i \leq M, , p \in \{1, 2\}$ as a representation of the probability of choosing item i from the set X under hypotheses G_p where $\gamma_{i_p} \in \{0, 1\}$. It further follows that $\sum_{i=1}^M \gamma_{i_p} = 1, p \in \{1, 2\}$, stipulating that one and only one of M items is selected.

Next, I introduce an observable random variable $Y \in \{1, 2, \cdots, N\}$, denoting the index of the bin (the proxy site) employed to carry one of the $M > N$ items. It follows that the probability of each bin's appearance given a set allocation scheme such as A_l under hypotheses G_p will be equal to

$$P(Y = j|A_l, G_p)$$

$$= \sum_{i=1}^{M} P(Y = j|A_l, X = i, G_p)P(X = i|A_l, G_p) \tag{4.2}$$

$$= \sum_{i=1}^{M} P(Y = j|A_l, X = i)P(X = i|G_p) \tag{4.3}$$

Furthermore, $P(Y = j|A_l, X = i) = \theta_{ij}^{(l)} \in \{0, , 1\}$. It thus follows that

$$P(Y = j|A_l, G_p) = \sum_{i=1}^{M} \theta_{ij}^{(l)} P(X = i|G_p) \rightarrow$$

$$P(Y = j|A_l, G_p) = \sum_{i \in S_j^{(l)}} \pi_{i_p} = \alpha_j^{(l,p)} \tag{4.4}$$

4.2.2 Revealed Information

By choosing to allocate M items to N bins where $N \leq M$, I have injected ambiguity and uncertainty into the output binning index sequence about the input item sequence over a successive n visits. In other words, if I originally chose to transmit n of such items, my total set of possible sequences would be of form $\overrightarrow{\mathbf{X^n}} = [X_1 X_2 ... X_n]$ out of M^n possible outcomes. From an observer's perspective which can only have access to which one of N bins is deployed in each time slot, sequences in the form of $\overrightarrow{\mathbf{Y^n}} = [Y_1 Y_2 ... Y_n]$ has cardinality of at most $N^n < M^n$. Despite the amount of uncertainty added due to the many-to-one mapping between items and bins, the output sequence sill reveals certain amount of information regarding the patterns of sequences of M random items.

This observation could be further studied by indicating how my allocation system resembles a coding framework where I have an equivalent channel whose input variable is X and output Y, as shown in Fig. 4.1.

Under such a framework, the equivalent channel output sequence $\overline{\mathbf{Y}}^n$ can help an eavesdropper classify the input sequence $\overline{\mathbf{X}}^n$ into a number of differential classes. As a result, information about the specific input item patterns is leaked to certain degree and can be measured using conditional mutual information $I(X;Y|A_l, G_p)$ between X and Y, under a hypotheses G_p given a particular binning (i. e. partition A_l relationship as illustrated in Fig. 4.1.

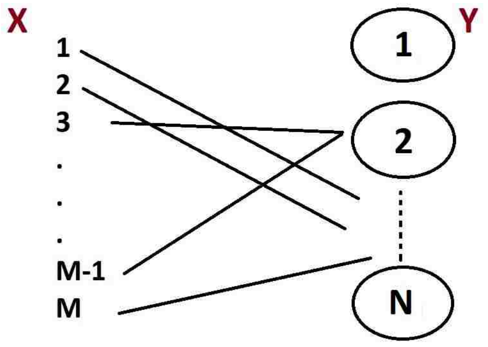

FIGURE 4.1. Coding Channel Representation of the Problem

Such conditional mutual information thus measures the maximum number of bits of meta information about item sequence per channel use. Therefore, I can have at most $2^{nI(X;Y|A_l,G_p)}$ sequences $\overline{\mathbf{X}^n}$ distinguishable by inferring based on $\overline{\mathbf{Y}^n}$.

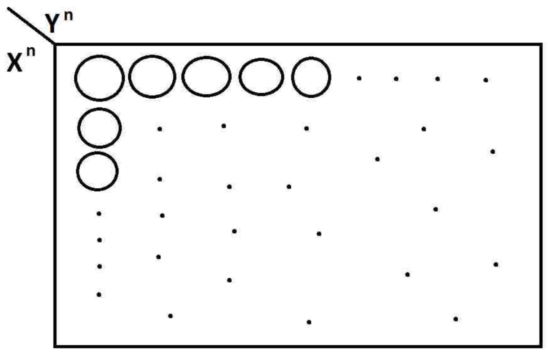

FIGURE 4.2. Jointly typical sequences

This concept could be further illustrated as shown in Fig.4.2 where a jointly typical set is presented [56]. There are about $2^{nH(X)}$ typical X sequences and about

$2^{nH(Y)}$ typical Y sequences. However, since there are only $2^{nH(X,Y)}$ jointly typical sequences, not all pairs of typical X^n and typical Y^n are also jointly typical. The probability that any randomly chosen pair is jointly typical is about $2^{-nI(X;Y)}$. Hence, I consider about $2^{nI(X;Y)}$ such pairs before I are likely to come across a jointly typical pair. This in turn, suggests that there are about $2^{nI(X;Y)}$ distinguishable signals X^n.

I thus adopt $I(X;Y|A_l, G_p)$ as the privacy metric conditioned on a particular partition mapping A_l under a specific hypotheses G_p. It follows that due to the combinatorial nature of a set allocation problem there are a total of N^M possible methods to allocate these M items to the N sets.

I can formulate the mutual information over a set allocation $A_l, 1 \leq l \leq N^M$ under hypotheses G_p as:

$$I(X;Y|A_l, G_p) = H(X|A_l, G_p) - H(X|Y, A_l, G_p) =$$
$$H(Y|A_l, G_p) - H(Y|X, A_l, G_p) = H(Y|A_l, G_p) =$$
$$H(\alpha_1^{(l,p)}, \alpha_2^{(l,p)}, ..., \alpha_N^{(l,p)}) \tag{4.5}$$

where I have used the notion of $H(a_1, a_2, ..., a_m) = -\sum_{v=1}^{m} a_v \log a_v$ and the fact that $H(Y|X, A_l, G_p) = 0$ because if the input X and the channel scheme A_l and hypotheses G_p are known, then output Y will offer no uncertainty.

4.2.3 Distinguishability Measure

In this section, I aim to define a measure which can evaluate the distinguishability between a number of discrete distributions G_p, with $p \in \{1, 2\}$.

Given that a certain set allocation A_l has been chosen, depending on the choice of G_p, two different distributions defined by using variables $\alpha_j^{(l,p)}$ could be developed where I have:

$$\alpha_j^{(l,p)} = P(Y = j|A_l, G_p), p \in \{1, 2\}, j \in \{1, ..., N\} \tag{4.6}$$

where the two probability sets $P_1 = \{\alpha_j^{(l,1)}, j = 1, 2, ..., N\}$ and $P_2 = \{\alpha_j^{(l,2)}, j = 1, 2, ..., N\}$ represent the distributions under allocation scheme l given two prior probability distributions G_1 and G_2 which I aim to distinguish. To evaluate the distinguishability between two such distributions, I utilize the definition of symmetrized KL divergence[57] defined as:

$$\begin{aligned} &\frac{1}{2}\{KL(P_1||P_2) + KL(P_2||P_1)\} \\ &= \frac{1}{2}\sum_{j=1}^{N}(\alpha_j^{(l,1)} - \alpha_j^{(l,2)})(\log \alpha_j^{(l,1)} - \log \alpha_j^{(l,2)}) \end{aligned} \tag{4.7}$$

It is important to note how this measure of distinguishability is different in nature than that of leaked information. Through binning, a sequence of outputs are produced which I aim to limit. However, I could use an average (note the average nature of KL distance) of these sequences over multiple choices as a measure of maximum likelihood detection deciding which hypotheses is truly active. By doing so, I am both controlling the level of leaked information and utilizing it for further gain.

4.2.4 Problem Formulation

Using the definitions addressed above, I could show that the UvP problem could be modeled in the form of finding the allocation scheme A_l which satisfies [49]

$$\boxed{\max_{0\leq l\leq N^M} \sum_{j=1}^{N}[\alpha_j^{(l,1)} f_j(S_j^{(l,1)}) + \lambda\alpha_j^{(l,1)}\log(\alpha_j^{(l,1)})]} \tag{4.8}$$

where $f_j(S_j^{(l,1)})$ represents the utility function offered by the j^{th} bin while using allocation scheme A_l under G_p hypotheses and $\lambda \geq 0$ demonstrates the Lagrange multiplier accounting for the leaked information constraint. Equation (4.8) is a sum of functions defined over a series of sets. I refer to these as multi-variate set functions seeing as how their values are based upon specific sets and variables introduced in each of these sets. Furthermore, I note that since there is only one active hypotheses G_1 in this scenario, I can have $\alpha_j^{(l,1)} = \alpha_j^{(l)}$ and $f_j(S_j^{(l,1)}) = f_j(S_j^{(l)})$.

I then aim aim to add another objective, namely, binary hypothesis testing between two candidate prior distributions $\{G_1, G_2\}$ on the item set. I thus choose binning as my method of observation meaning I attempt to find an M-to-N mapping between the set of M items and set of N outputs which can best help us distinguish which distribution was chosen.

Overall, I aim to find a partition of M items under which I can maximize the binning utility of the solution while keeping the distinguishability between G_1 and G_2 and the leaked information about the item sequences higher and less than some thresholds respectively.

$$\boxed{\begin{aligned} \max_{1\le l\le N^M} \quad & \sum_{p=1}^{2}\sum_{j=1}^{N} P(G_p)[\alpha_j^{(l,p)} f(S_j^{(l,p)}) + \lambda_2 \alpha_j^{(l,p)} \log \alpha_j^{(l,p)}] \\ & + \frac{\lambda_1}{2}\sum_{j=1}^{N} (\alpha_j^{(l,1)} - \alpha_j^{(l,2)})(\log \alpha_j^{(l,1)} - \log \alpha_j^{(l,2)}) \qquad (4.9) \end{aligned}}$$

I thus pose my problem as a constrained optimization problem with the goal of maximizing the average binning utility while controlling the average leakage and measure of distinguishability, as reflected by the range of some regularization terms in Eq.(4.9) where $\lambda_1, \lambda_2 > 0$ are Lagrange multipliers accounting for the constraints over distinguishability and leaked information respectively.

4.3 Multi-Submodular Set Functions as a Means of Solution

As mentioned previously, both UvP and UvPD are NP-complicated (they are solved when searches over N^M and 2^M possible set allocations are carried out respectively and the best allocation is chosen). Still, I could opt to utilize the definition of multi-submodular set functions so as to reduce the complexity to that of polynomial at the cost of accuracy.

Note: For any further references, I first need to address a series of variable and function definitions which are going to play a vital role in the rest of this chapter:

Definitions for Problem 1

1. Any variable represented with a capital Letter represents a set.
2. Any variable represented with a small letter represents an element.
3. $A - B$ represents a set containing all elements of set A which do not appear in set B.
4. α_x represents the probability of item x and α_A represents the sum of probabilities of items mapped into a set A.
5. α_{BA} represents the difference in the sum of probabilities of items mapped into the sets B and A which could be further shown as $\alpha_{BA} = \alpha_B - \alpha_A$.
6. $g(C, D)$ represents the 1^{st} order difference of a set function $f(C)$ from $f(C-D)$ where $D \subseteq C$ which could be formulated as $g(C, D) = f(C) - f(C - D)$.
7. $q(C, C_1, D, D_1)$ represents the 2^{nd} order difference of a set function $f(C)$ where $C_1 \subseteq C$ and $D_1 \subseteq D$ which could be formulated as $q(C, C_1, D, D_1) = g(C, D) - g(C_1, D_1)$.
8. I assume the probability of items is sorted in a decreasing manner such as $\pi_1 \geq \pi_2 \geq ... \geq \pi_M$.

4.3.1 Imposing Multi-submodularity

In [42], it was shown that if I can prove multi-submodularity of complex set functions, then they could be modeled as simpler problems (submodular set functions). I thus, aim to find the sufficient conditions for such occurrence. To do so, I first offer a brief review of multi-submodularity.

As in [42], if I define $\mathbb{M} = \{1, 2, ..., M\}$, then a multivariate function $F : (2^{\mathbb{M}})^N \rightarrow \mathbb{R}^+$ is multi-submodular if for all pairs of tuples $(S_1, ..., S_N)$ and $(T_1, ...T_N) \in (2^{\mathbb{M}})^N$

I will have:

$$F(S_1, ..., S_N) + F(T_1, ..., T_N) \geq F(S_1 \cup T_1, ..., S_N \cup T_N)$$
$$+F(S_1 \cap T_1, ..., S_N \cap T_N) \tag{4.10}$$

In my formulation of the problem, on the other hand, the function $F(S_1, S_2, ..., S_N)$ is defined as

$$\sum_{j=1}^{N} [\alpha_j^{(l,1)} f_j(S_j^{(l,1)}) + \lambda \alpha_j^{(l,1)} \log(\alpha_j^{(l,1)})] \tag{4.11}$$

where the two components $\alpha_j^{(l,1)} f_j(S_j^{(l,1)})$ and $\alpha_j^{(l,1)} \log(\alpha_j^{(l,1)})$ represent the components carrying the effect of one set $S_j^{(l,1)}$. As a result, Eq.(4.10) can be further expressed as

$$\begin{aligned}
&\{\alpha_1 f(S_1) + \lambda\alpha_1 \log \alpha_1\} + \{\alpha_2 f(S_2) + \lambda\alpha_2 \log \alpha_2\} + ... \\
&+\{\alpha_N f(S_N) + \lambda\alpha_N \log \alpha_N\} + \{\beta_1 f(T_1) + \lambda\beta_1 \log \beta_1\}+ \\
&\{\beta_2 f(T_2) + \lambda\beta_2 \log \beta_2\} + ... + \{\beta_N f(T_N) + \lambda\beta_N \log \beta_N\} \\
&\geq \{\omega_1 f(S_1 \cup T_1) + \lambda\omega_1 \log \omega_1\} \\
&+... + \{\omega_N f(S_N \cup T_N) + \lambda\omega_N \log \omega_N\} \\
&+\{\zeta_1 f(S_1 \cap T_1) + \lambda\zeta_1 \log \zeta_1\} \\
&+... + \{\zeta_N f(S_N \cap T_N) + \lambda\zeta_N \log \zeta_N\}
\end{aligned} \tag{4.12}$$

where $\alpha_j, \beta_j, \omega_j$ and ζ_j represent the probability of output $Y = 1$ when the allocation schemes $S_j, T_j, S_j \cup T_j$ and $S_j \cap T_j$ are respectively chosen.

It follows that one sufficient condition to ascertain that Eq.(4.12) holds true would be to satisfy submodularity for each function f over any set meaning I find the sufficient conditions that result in

$$\{\alpha_j f(S_j) + \lambda\alpha_j \log\alpha_j\} + \{\beta_j f(T_j) + \lambda\beta_j \log\beta_j\} \geq$$
$$\{\omega_j f(S_j \cup T_j) + \lambda\omega_j \log\omega_j\} + \{\zeta_j f(S_j \cap T_j)$$
$$+\lambda\zeta_j \log\zeta_j\}, \quad \forall j \in \{1, .., N\} \tag{4.13}$$

then, Eq.(4.10) will be satisfied and thus the overall function will be multi-submodular.

Satisfying Eq.(4.13) would be equivalent to making sure that a set function $G(S)$ under any possible set allocation A_l as defined below

$$G(S) = [\alpha^{(l,1)} f(S^{(l,1)}) + \lambda\alpha^{(l,1)} \log(\alpha^{(l,1)})] \tag{4.14}$$

is submodular.

4.3.2 Imposing Separate Submodularities for UvP

For an easier mathematical representation of the following derivation I denote $G(S_j) = G_j^{(l)}$. Furthermore, I denote $f_j(S_j^{(l)}) = f(S_j^{(l)})$. Both these denotations allude to the fact that once a set allocation A_l is chosen, its index could be dropped.

In the next step, I opt to use diminishing return property as the means of making certain each of these functions are submodular. Following is a definition of diminishing returns for submodular functions, after which I derive the sufficient conditions for the case discussed in UvP).

Diminishing Property Return dictates that if I define $\mathbb{S}$ as the universal set, a set function $F : 2^{\mathbb{S}} \rightarrow \mathbb{R}^+$ is submodular if, for all $A, B \subseteq \mathbb{S}$ with $A \subseteq B$ and for each $x \in \mathbb{S} - B$ I have [51]:

$$F(A \cup \{x\}) - F(A) \geq F(B \cup \{x\}) - F(B) \tag{4.15}$$

Now I attempt to expand Eq. (4.15) for each $G(S_j)$. However, to properly do so, I first need to account for the behavior of this function. While thus far, I have found Eq.(4.13) to be a sufficient condition of multisubmodularity on F, there is another issue I need to address which comes from the finite nature of the universal set in my problem settings.

In my problem it is assumed that $S_1 \cup S_2 \cup ... \cup S_N = T_1 \cup T_2 \cup ... \cup T_N = \{1, 2, .., M\} = U$. In other words, any of the sets $S_j, j \in \{1, .., N\}$ can be described as

$$S_j = U - \uplus_{i \neq j, i=1}^{N} S_i \tag{4.16}$$

This in turn means that any set S_j can be modeled as a complement of other sets in the same settings and as a result, in order to ensure multi-submodularity, I need to ensure that the submodularity condition for set functions of type Eq.(4.13) also hold true for a complementary set S^C which in turn results in a second sufficient condition:

$$\begin{gathered}
\{(1-\alpha_j)f(U-S_j) + \lambda(1-\alpha_j)\log(1-\alpha_j)\} \\
+\{(1-\beta_j)f(U-T_j) + \lambda(1-\beta_j)\log(1-\beta_j)\} \geq \\
\{(1-\omega_j)f(U-\{S_j \cup T_j\}) + \lambda(1-\omega_j)\log(1-\omega_j)\} \\
+\{(1-\zeta_j)f(U-\{S_j \cap T_j\}) + \lambda(1-\zeta_j)\log(1-\zeta_j)\}, \\
\forall j \in \{1, .., N\}
\end{gathered} \tag{4.17}$$

This condition, which has been referred to as duality in my definitions, guarantees that the dual nature of the sets (the fact that they are defined over a finite universal set) does not cause any problems for my set of sufficient conditions.

Thus, for any set S_j, I must find the sufficient conditions for the existence of diminishing property for both functions $G_1(S_j) = \alpha_j f(S_j) + \lambda\alpha_j \log(\alpha_j)$ and $G_2(S_j) = (1-\alpha_j)f(S-S_j) + \lambda(1-\alpha_j)\log(1-\alpha_j)$. To do so, I will evaluate their sufficient conditions and then find their intersection as the final conditions (assuming they do not negate one another).

Theorem 4.3.1. *For a general $N \geq 2$, the set functions $G_1(S_j)$ and $G_2(S_j)$ and as a result $G(S_j)$ are submodular if*

(1) $g(S_j, S_w) \leq 0$

(2) $q(S_j, S_u, S_w, S_y) \leq 0$

(3) $|g(S_j, S_w)| \geq \lambda \log\left(\frac{1}{\pi_M}\right)$

for all possible sets $S_w \subseteq S_j \subseteq S$ and $S_u \subseteq S_j$ and $S_y \subseteq S_w$ where S is the universal set.

The proof for this theorem is presented in the Appendix under Theorem III. 1.

Thus far, I have found the sufficient conditions of submodularity for a binning problem where $N \geq 2$ using previous works. It would be interesting to see how such conditions change when I take into account the specific case of $N = 2$ and there is a symmetry between the 2 bins.

4.3.3 Specific Case of $N = 2$

As mentioned previously, in order to show the applicability of submodular functions I choose to reiterate the overall utility function dictated in UvP for when $N = 2$:

$$\begin{aligned}\max_{1 \leq l \leq 2^M} \quad & \alpha_1^{(l)} f(S_1^{(l)}) + (1 - \alpha_1^{(l)}) f(S - S_1^{(l)}) \\ & + \alpha_1^{(l)} \log(\alpha_1^{(l)}) + (1 - \alpha_1^{(l)}) \log(1 - \alpha_1^{(l)})) \\ & = T(S_1) \end{aligned} \tag{4.18}$$

As can be seen, the problem is still exponentially complex seeing as how I need to search over 2^M possible solutions to find the optimal. Thus, once again I aim to impose multi-submodularity (in this case simplified to submodularity) on the new utility function. For the utility function above the same results derived for a general

N could be used as a set of sufficient conditions. However, taking into account the joint relationship between the 2 sets and writing the same Inequality (4.15) for Equality (4.18) I am able to find a less restrictive set of sufficient conditions for the submodularity of this utility function as indicated below:

Theorem 4.3.2. *When $N = 2$, the function in Eq. (4.18) is submodular if*

(1) $g(S_j^{(l)}, S_w^{(l)}) \leq 0$

(2) $2|g(S_j^{(l)}, S_w^{(l)})| \geq \lambda \log(1 + \frac{1-\pi_M}{\pi_M^2})$

(3) $q(S_j^{(l)}, S_u^{(l)}, S_w^{(l)}, S_y^{(l)}) \leq 0$

for all possible sets $S_w^{(l)} \subseteq S_j^{(l)} \subseteq S$ *and* $S_u^{(l)} \subseteq S_j^{(l)}$ *and* $S_y^{(l)} \subseteq S_w^{(l)}$ *where* S *is the universal set.*

where I have deduced that the new set of sufficient conditions are less restrictive because they allow a smaller lower bound (a looser lower bound) for the absolute value of the first order derivative of the set function $f(S_j)$.

The proof for this theorem is presented in Appendix under Theorem III. 2 in [8]. The steps in this theorem are quite close to those taken in the proof for Theorem III.1 where the only difference is in how I utilize the dual nature of the formulation as developed in Eq.(4.18) to further loosen the sufficient conditions.

4.3.4 Imposing Separate Submodularities for the UvPD Problem

In this section, I aim to find sufficient conditions for multisubmodularity of the overall utility function described for the UvPD Problem.

Once again, I use the intuition offered in the previous section to break down Eq. (4.9) into a sum of simpler set functions and then impose submodularity over all such sets and thus guarantee the overall multisubmodularity as well. Following the previous reasoning, I aim to find the sufficient conditions of submodularity for

$$\begin{aligned} G(S_j^{(l)}) = P(G_1)[\alpha_j^{(l,1)} f(S_j^{(l,1)}) + \lambda_2 \alpha_j^{(l,1)} \log \alpha_j^{(l,1)}] \\ +P(G_2)[\alpha_j^{(l,2)} f(S_j^{(l,2)}) + \lambda_2 \alpha_j^{(l,2)} \log \alpha_j^{(l,2)}] \\ +\frac{\lambda_1}{2}(\alpha_j^{(l,1)} - \alpha_j^{(l,2)})(\log \alpha_j^{(l,1)} - \log \alpha_j^{(l,2)}) \end{aligned} \tag{4.19}$$

for all sets S_j. Assuming that a specific allocation scheme A_l has been chosen, I can further simplify Eq. (4.19) by rewriting

$$\begin{aligned} G(S_j^{(l)}) = G(S_j) = P(G_1)[\alpha_j^{(1)} f(S_j^{(1)}) + \lambda_2 \alpha_j^{(1)} \log \alpha_j^{(1)}] \\ +P(G_2)[\alpha_j^{(2)} f(S_j^{(2)}) + \lambda_2 \alpha_j^{(2)} \log \alpha_j^{(2)}] \\ +\frac{\lambda_1}{2}(\alpha_j^{(1)} - \alpha_j^{(2)})(\log \alpha_j^{(l,1)} - \log \alpha_j^{(l,2)}) \end{aligned} \tag{4.20}$$

where I have simply dropped every iteration of l from my previous formulation for easier representation.

As in the previous cases, it seems that the function $G(S_j)$ has a singular relationship with set S_j. However; there is a secondary relationship the function shares with the set $S_j^{\mathbf{C}} = S - S_j$ where $S = \mathbb{S}$ represents the universal set. This relationship could be modeled as

$$\begin{aligned} G(S_j^C) = P(G_1)[(1 - \beta_j^{(l,1)}) f(S - S_j^{(l,1)^C}) \\ +\lambda_2 (1 - \beta_j^{(l,1)}) \log (1 - \beta_j^{(l,1)})] \\ +P(G_2)[(1 - \beta_j^{(l,2)}) f(S - S_j^{(l,2)^C}) \\ +\lambda_2 (1 - \beta_j^{(l,2)}) \log (1 - \beta_j^{(l,2)})] \\ +\frac{\lambda_1}{2}(\beta_j^{(l,2)} - \beta_j^{(1)})(\log (1 - \beta_j^{(1)}) - \log (1 - \beta_j^{(2)})) \end{aligned} \tag{4.21}$$

where I have used the fact that $\beta_j = 1 - \alpha_j$ seeing as how I define

$$\beta_j^{(l)} = \sum_{i \in \{S - S_j^{(l)}\}} \pi_i \tag{4.22}$$

Thus, for any set S_j, I must find the sufficient conditions for the existence of diminishing property for both functions described in Eq. (4.20) and Eq. (4.21). To do so, I will evaluate their sufficient conditions and then find their intersection as the final conditions (assuming they do not negate one another).

Theorem 4.3.3. *The set functions defined for the UvPD for a general $N \geq 2$ problem is submodular if*

$$\begin{gathered}(1) \quad q(S_j, S_w, S_r, S_t) \leq 0 \\ (2) \quad g(S_j, S_w) \leq 0 \\ (3) \quad |g(S_j, S_w)| \geq \\ max\{\lambda_2 \log \omega_1' + \frac{\lambda_1}{P(G_1)} \log(\omega_1' \omega_2'), \\ \lambda_2 \log \omega_2' + \frac{\lambda_1}{P(G_2)} \log(\omega_2' \omega_1')\} \\ , \omega_1' = \frac{1}{\pi_{M_1}}, \omega_2' = \frac{1}{\pi_{M_2}}\end{gathered} \tag{4.23}$$

for all possible sets $S_w \subseteq S_j$ and $S_t \subseteq S_r$ and $S_r \subseteq S_j$.

The proof for this theorem is presented in the appendix under Theorem III. 3 in [8]. Here, a series of sufficient conditions for $G(S_j)$ as formulated in either Eq. (4.20) or in Eq. (4.21) are evaluated separately and their intersection is calculated.

It is important to note how adding distinguishability as a factor to account for has changed the dynamic of my sets of sufficient conditions to assure accessibility of a submodular solution for the problem. In the UvP formulation of the problem, λ referred to the Lagrange multiplier accounting for the leaked information constraint. The same role is played by λ_2 in the formulation of UvPD problem. Furthermore, in both problems, I aim to maximize a utility while keeping the leaked information to a threshold. This would mean that if there were no concerns of distinguishability in the UvPD problem, I would be expecting the same sets of sufficient condition as the two problems basically become the same. As can be seen, if I assume λ_1 (which represents the Lagrange multiplier accounting for the distinguishability measure) is very small, then the two sets of sufficient conditions turn out to be the same (noting that there is basically only one hypotheses to account for when distinguishability is of no importance).

4.3.5 Selection of Utility Function

In this section I aim to detail how to use the results gathered in previous theorems and Algorithms to find a proper utility function given the specific properties of the problem. I assume a fixed arbitrary value of M and $N = 2$ and randomly generate item probabilities all the while assuming that $\pi_{M_1}, \pi_{M_2} \geq \delta$. I then use $P(G_1)$ and $P(G_2) = 1 - P(G_1)$ and fixed values of λ_1, λ_2 and λ. Finally, I assume that $f(S) = f(|S|)$ where S and $|S|$ represent any set and its cardinality respectively.

It is important to note that in my derivations of a desirable utility function I focus on developing a quadratic utility function. Such an assumption might appear to be extremely limiting to my class of functions at first sight. However, as is further explained in [44], such an assumption is quite understandable and rather desirable seeing as how in many economic models, functions are written as extensions of quadratic function and thus later calculations are simplified while still maintaining the essence of a utility function. Furthermore, such an assumption for the set function will result in the simplification of 1^{st} and 2^{nd} order derivatives of set function f into linear and constant functions respectively.

Here I try to find a utility set function which can satisfy the sufficient conditions gathered in Theorems III.1 and III.3 for UvP and UvPD respectively. In order to save space, I choose to find a utility function which can satisfy both the sufficient conditions for UvP and UvPD's multisubmodularity. Thus, the objective functions should satisfy the first 2 conditions from Theorems III.1 and III.2 and the following sufficient condition:

$$|g(S_j, S_w)| \geq Q \tag{4.24}$$

where I have defined

$$Q = max\{\lambda_2 \log \omega_1' + \frac{\lambda_1}{P(G_1)} \log (\omega_1' \omega_2'),$$
$$\lambda_2 \log \omega_2' + \frac{\lambda_1}{P(G_2)} \log (\omega_2' \omega_1'), \frac{\lambda}{2} \log (1 + \frac{1-\delta}{\delta^2})\} \tag{4.25}$$

and

$$\omega_1' = \omega_2' = \frac{1}{\delta} \tag{4.26}$$

As previously mentioned, I assume that $f(S) = a|S|^2 + b|S| + c$. Thus, $g(S) = 2a|S| + (b - a)$ and $q(S) = 2a$. The above equations could thus be rewritten as:

$$\begin{gathered} (1) \quad 2a \leq 0 \\ (2) \quad 2a|S| + b - a \leq 0 \\ (3) \quad |2a|S| + b - a| \geq Q \quad \forall |S| \in \{1, ..., M\} \end{gathered} \tag{4.27}$$

I further choose to add another condition $f(S) \geq 0 \quad \forall S$ which ascertains that the cost function is always positive.

I can thus plot the above 4 conditions in a 3-D figure consisting of a, b and c and find the region acceptable for all 4 conditions resulting in a range of acceptable a, b and c's.

An Example of such derivation will be offered in Section 4.5.

4.4 Submodular Solution

In this section, I offer two algorithms specifying how each UvP and UvPD problem could be solved in a polynomial manner (assuming $N = 2$). I further expand upon how the complexity solutions has been compromised. Finally, I offer an example to showcase the relationship between solution error and complexity.

4.4.1 Solution to UvP

I choose to modify the results from [58] to create my own algorithm to find the submodular function solution to my problem. I present:

```
Algorithm 1: Submodular Function Solution to the UvP Problem)
  1. Let $S_1 = argmax_{e \in X = \{1,...,M\}} T[S_1 = \{e\}]$.
  2. If there is an element $e \in X \setminus S_1$ such that $T[S_1 + \{e\}] \geq T[S_1]$, let $S_1 = S_1 + \{e\}$.
  3. If there is an element $e \in S_1$ such that $T[S_1 \setminus \{e\}] \geq T[S_1]$, let $S_1 = S_1 - \{e\}$.
     Go to Step 2.
  4. Return maximum of $T[S_1]$ and $T[X \setminus S_1]$.
```

4.4.2 Solution to the UvPD Problem

Following the same pattern as that of Algorithm 1, I can write:

```
Algorithm 2: Submodular Function Solution to the UvPD Problem
  1. Let $S_1 = argmax_{e \in X = \{1,...,M\}} T[S_1 = \{e\}]$.
  2. If there is an element $e \in X \setminus S_1$ such that $T[S_1 + \{e\}] \geq T[S_1]$, let $S_1 = S_1 + \{e_1\}$.
  3. If there is an element $e \in S_1$ such that $T[S_1 \setminus \{e\}] \geq T[S_1]$, let $S_1 = S_1 - \{e\}$.
     Go to Step 2.
  4. Return maximum of $T[S_1]$ and $T[X \setminus S_1]$.
```

In both cases, I know that at the very last step $T[S_1] = T[X \setminus S_1]$. Now I opt to calculate the complexities of this method. Steps 2 and 3 could repeat $(M-1)+(M-2)+...+1 = \frac{M(M+1)}{2}$ times each while every item could be removed and thus replaced a total of $2M$ times. Thus the total complexity of steps 2 and 3 is equal to $M^2(M+1) = O(M^3)$. The complexity of step 1 is also equal to M. Thus the total complexity of the solution is equal to $O(M^3)$.

These polynomial solutions simply make certain the maximal and minimal functions obtained are at least 0.432 and at most 2.315 times the optimal objective functions respectively. This range of error occurs because in this method, I am removing and adding members from and to the set S_1 one by one. Thus, at each decision point I am making one locally optimal decision. However, it is widely known that a locally greedy method is not necessarily globally optimal [55].

I acknowledge that the algorithms proposed above represent a modification of the original set-allocation algorithm proposed in [58] with $N = 2$ for submodular problems. It's important to note that for any $N > 2$ the algorithm could get extremely complicated as in $N = 2$ the addition of an item to a set S_1 means the removal of the same item from the other set S_2 while in more general values of N such a relationship is not necessarily true and thus many more complicated scenarios need to be considered. As a result, I have decided to offer results for the case of $N = 2$ (two bins) to demonstrate the validity of of my proved sufficient conditions which enable use to modify the existing algorithms in my scenarios.

4.5 Numerical Examples

In this section I aim to offer examples where I aim to use the results gathered in previous theorems and Algorithms to first find a proper utility function given the specifics and then run and test the Algorithms' results with that of an exhaustive search to compare the two methods in terms of complexity and exactness of the solution. To address a variety in my results, I assume different values of $M = 6, ..., 20$ and randomly generate item probabilities all the while assuming that $\pi_{M_1}, \pi_{M_2} \geq 10^{-5}$ due to MATLAB settings. I then use $P(G_1) = 0.8$ and $P(G_2) = 0.2$ and $\lambda = \lambda_2 = 0.4, \lambda_1 = 0.6$ and that $f(S) = f(|S|)$ where S and $|S|$ represent any set and its cardinality respectively to find appropriate utility function to make certain the submodular solution is converging and relatively exact.

It is important to note that my main goal here is to guarantee a polynomial algorithm to UvP and UvPD problems is relatively acceptable in comparison with an NP-complicated exhaustive algorithm. I thus choose to simply plot the final objective functions of UvP and UvPD algorithms and compare them with one another in Fig.4.4 and 4.5. Using the previously developed results, it could be seen that I need to satisfy conditions (1) and (2) from Eq.(4.27) and a third condition

$$|2a|S| + b - a| \geq 73.66 \quad \forall |S| \in \{5, 6, ..., 20\} \tag{4.28}$$

The condition (2) from Eq.(4.27) and the condition from Eq.(4.28) can be plotted in Fig.(4.3) where in the top figure, the allowed region is on the left side of the border defined by the many graphs and in the bottom figure, the allowed region is on the outer side of the borders indicated by the many graphs. I also have a third condition which entails that $a \leq 0$ I need to tend to.

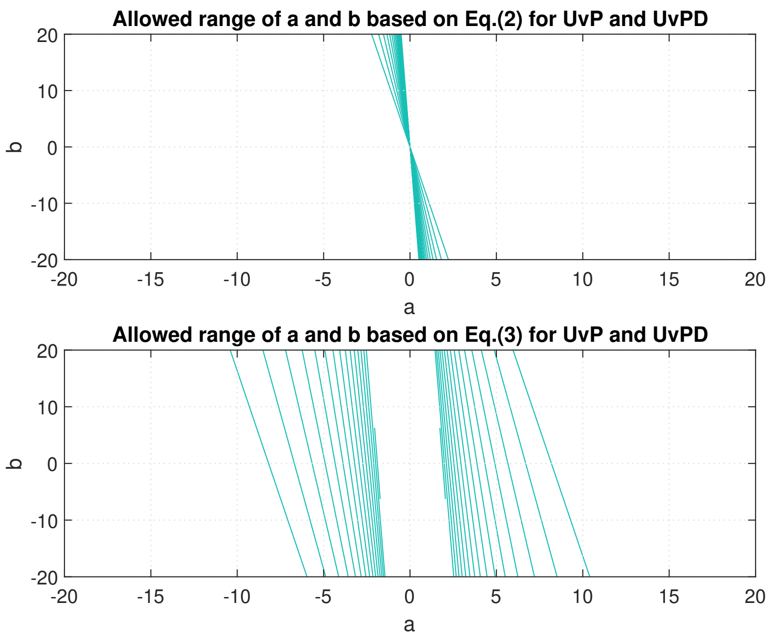

FIGURE 4.3. a and b regions based on different sufficient conditions

Taking all these conditions into account one possible value for a and b could be $a = -15, b = 10$. Finally, in order to account for the value of c, I note that

$f(S) \geq 0 \quad \forall S$ meaning

$$\begin{aligned} -15|S|^2 + 10|S| + c \geq 0 \quad \forall |S| \in \{5, 6, ..., 20\} \\ \rightarrow c \geq 15(|S| - \frac{1}{3})^2 - \frac{5}{3} \\ \rightarrow c \geq 5802 \end{aligned} \tag{4.29}$$

Finally, the utility function is defined to be $f(S) = -15|S|^2 + 10|S| + 5810$.

4.5.1 Comparison for UvP

The results of running Algorithm 1 and an exhaustive search algorithm on different UvP problems have been gathered in Fig.4.4 .

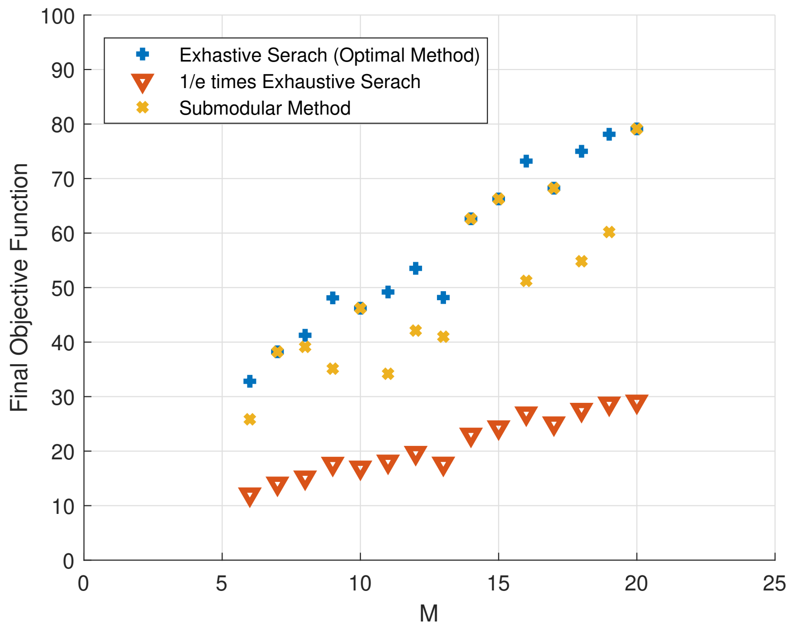

FIGURE 4.4. Objective Functions for UvP through Different Algorithms

As can be seen, there are times when the submodular solution has the same optimal output as those of exhaustive search and times when it showcases a loss of exactness. Although I previously mentioned the possibility of a loss of accuracy, I still need to discuss the reason behind this disparity of results. While running the submodular algorithm for some examples, I need to satisfy maximization of the utility function, this then results in the selection of the item with highest allowed probability. However, once this item is fixed into the solution set, I cannot add any more items, because adding any further items will result in a lesser overall objective function. However, if I was to run the exhaustive search method, I would see that choosing the two items with the lowest probabilities would have offered a better overall utility. Thus, the biggest limitation in submodular set solutions appears in the process of allocating the first item to the solution set. Depending on different scenarios, such an issue may or may not rise.

4.5.2 Comparison for UvPD

Once again, assuming $M = 6, ..., 20$, I offer multiple examples of the performance of submodular solution in the UvPD problem with a further assumption that $P(G_1) = 0.8$ and $P(G_2) = 0.2$. The results have been gathered in Fig.(4.5).

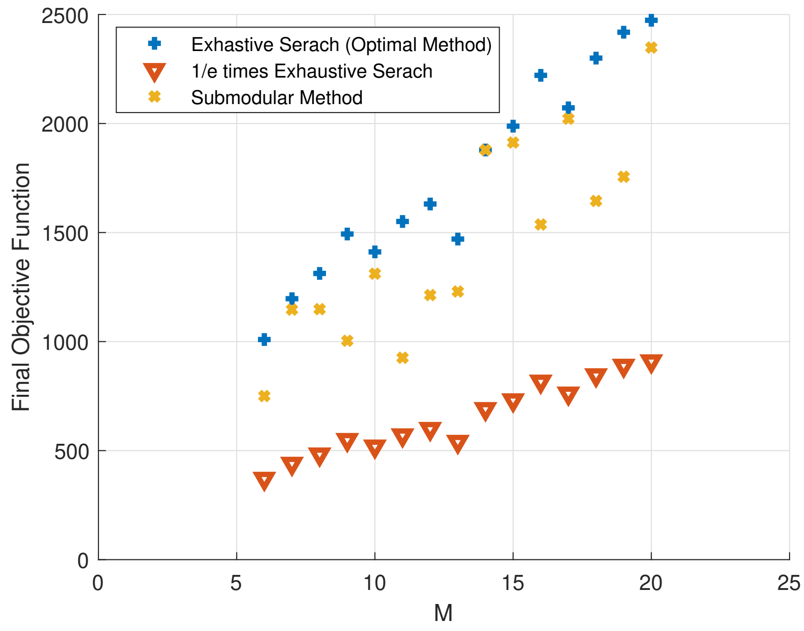

FIGURE 4.5. Objective Function for UvPD through Different Algorithms

As can be witnessed, due to the addition to hypotheses probabilities $P(G_1)$ and $P(G_2)$ and their inclusion in the conditions gathered in Theorem III.3, the utility functions adjusted for submodularity in UvPD are much larger than those appearing in UvP. As a result, the overall objective function is also much larger than the one gathered in Fig.(4.4). Otherwise, the results in comparison with those gathered by exhaustive search are quite similar to those seen previously in Fig.(4.4).

Chapter 5
Information Bottleneck Problem Revisited

5.1 Introduction

The information bottleneck problem was first introduced as a trade-off problem between an input and an output through a median variable [9]. Namely, given two random variables U and X following a given joint distribution $P(U, X)$, I seek a mapping between X and Y to form a Markov Chain $U \rightarrow X \rightarrow Y$ under which the mutual information between X and Y is minimized under a constraint that the mutual information between U and Y is higher than a lower bound. A straightforward instance for such a model could be exemplified when I am observing X at one point, which shall be quantized and shared with another entity, who is interested in inferring about U using such quantized information of X. Thus, the cost due to distortion induced by compression is in terms of mutual information $I(U; Y)$. In this scenario, I hope that minimal rate is used to describe X through output Y, however I also want to make sure that an enough level of information about input U could be revealed through output Y.

In the information bottleneck method as detailed by [9], the above constrained optimization was posed as a non-convex problem with remarks of lacking a guaranteed globally optimal solution. The problem was then rewritten in the format of a Lagrange multiplier problem where the goal would be to minimize a linear combination of the two functions ($I(X; Y)$ and $I(U; Y)$) connected to one another through a Lagrange multiplier.

Such interpretation of the problem gave way to the use of convenient Lagrange multiplier methods and an iterative algorithm with every step of the algorithm specified in a simple formula. It was shown in [9] that the solution might not be optimal and instead they emphasized on the importance of the simplicity of the algorithm. This simplicity has resulted in the Information Bottleneck (IB) algorithm being widely used in many settings including both learning driven and information theoretic problems.

In this chapter, I revisit the original information bottleneck problem in a constrained optimization setting. I go into details as to what the concerns of using the simplified Lagrange method are and not only offer a new algorithm of solving the problem to address such issues, but also (by going into details about the potential issues in the canonical IB approach) further justify the adoption of free and auxiliary variables as described in my new algorithm.

We then propose to adopt a more systematic method in optimization literature, namely Alternate Direction Method of Multipliers (ADMM) to develop a more efficient algorithm to solve such problems. This method offers the option of adapting penalty functions as a means of controlling the constraints imposed on the problem [12]. I showcase how such a method is superior to previous algorithms by both discussing the nature of two solutions and offering sufficient numerical evidence.

5.1.1 Related Works

Due to the nature of my chapter, which heavily relies on discussing every step of the information bottleneck method, my main point of reference will be the original Information Bottleneck paper [9]. We later find it important to note that [9] was inspired by the works of [10] and [11]. Thus, from time to time, in order to indicate the inadequacies of [9]'s results, I will refer to these papers. Further comments on the inadequacies of [9] have been made in many works as recent as [15] which reflect the concerns of many previous authors who while aware of the limitations of IB, still chose to utilize it due to its simple implementation.

The idea of solving Lagrange multiplier problems where the goal is to either maximize or minimize a function using penalty functions has been explored in many previous mathematical and computing works such as [12]. These issues could be modeled in the form of an Augmented Lagrange Multiplier (ALM) problem. However, due to the still-complex nature of solving any non-convex Lagrange multiplier problem (which requires carrying out gradient descent and checking for conditions on every Lagrange multiplier), further studies and methods are required. One algorithm of dealing with the limitations of an ALM problem is to instead try and adopt an ADMM algorithm. One of the most recent works about this method is [13] which details the superiority of adopting ADMM over continuing with ALM. [13] makes further observations on the accuracy of a non-convex Augmented Lagrange Multiplier problem which I will utilize to justify the numerical results achieved through this chapter.

Finally, [48] offers a series of sufficient conditions on the desired utility function by which a convergence of the ADMM method could be guaranteed. I will offer insight into these sets of sufficient conditions and whether or not they are applicable to my problem settings.

5.1.2 chapter Contributions

In this chapter, I (1) revisit the Information Bottleneck problem and the algorithm to solving it as suggested by [9]; (2) explain exactly if and how any step in the original algorithm could be problematic; (3) introduce the concept of augmented Lagrange multiplier problems to the same original problem; (4) showcase every possible superiority offered by the new method of looking at the problem; (5) offer an algorithm and an in-depth look at how it could be implemented to my problem; (6) demonstrate the practical results of running my suggested algorithm (ADMM) over the same problem; and (7) offer in-depth reasoning for the numerical results to further justify the novelty and importance of my new suggestion.

Finally, some proof and observations have been left out of this work and instead further explained in the technical report [59] for any required clarification.

5.2 System Model and Problem Formulation

We assume a Markov chain $U \to X \to Y$ meaning

$$p(Y = y|U = u, X = x) = p(Y = y|X = x) \tag{5.1}$$

Then, I am seeking a conditional PMF $P_{Y|X}(y|x)$ to attain an optimal tradeoff between quantization rate and information loss about a hidden variable U. I could rewrite this problem in the following manner:

$$\begin{aligned} min_{P_{Y|X}(y|x)} \quad & I(X;Y) \\ s.t \quad & I(U,Y) \geq I_{th} \end{aligned} \tag{5.2}$$

I will later address how the problem formulated in Eq.(5.2) represents a non-convex optimization problem and what this classification means for my studies. However, for the time-being, I choose to rewrite Eq.(5.2) in the format of a Lagrange multiplier problem:

$$min_{P_{Y|X}(y|x)} \quad I(X;Y) - \beta I(U;Y) \tag{5.3}$$

where $\beta \geq 0$ is assumed to be a Lagrange multiplier connecting the two mutual information functions given to us by the user and I have removed the term $+\beta I_{th}$ from the utility function I aim to minimize as it would be a constant for a fixed value of β.

Furthermore, there are a series of hidden constraints imposed upon the problem which ensure that the final solution $P_{Y|X}(y|x)$ is a probabilistic matrix resulting in:

$$\begin{aligned} &(1) \quad p(y|x) \in [0,1] \forall x \in \mathcal{X}, y \in \mathcal{Y}, \\ &(2) \quad \sum_{y \in Y} p(y|x) = 1, \forall x \in \mathcal{X} \end{aligned} \tag{5.4}$$

These sets of conditions, while important, could be dealt with in two different manners; one as suggested by [9] and the other as suggested in this chapter. As a result, I instead change my focus to the problem formulated in either Eq.(5.2) or Eq.(5.3) and only address the conditions in Eq.(5.4) when needed.

5.3 Revisiting the Original Information Bottleneck Solution

In this section, I briefly revisit the solution method offered by [9] to exemplify how the problem was tackled previously and so as to offer an in-depth study of the possible limitations of such an algorithm.

To do so, I find it necessary to introduce two types of variables: (1) Free variables which represent the set of variables I am able to choose so as to optimize the overall objective function; in other words, they are the variables achieved by imposing the first order derivative to be equal to 0; (2) Auxiliary variables which help form an iterative relationship with free variables resulting in a more methodical algorithm.

5.3.1 Problem Formulation

In [9], it was first acknowledged how the problem formulated in Eq.(5.2) is about minimizing a convex function over a non-convex set. Thus, the overall problem is a non-convex optimization. The proof, while not necessarily difficult, was never fully presented. Here I offer the proof for this observation:

Theorem 5.3.1. *The optimization problem presented in Eq.(5.2) represents a non-convex optimization.*

The proof for this theorem is presented in Appendix under Theorem III.1 in [59]. To prove the non-convexity of the optimization, I first show that the functions

$I(X;Y)$ and $I(U;Y)$ are both convex functions of $p(y|x)$. Then it could be argued that the set $I(U;Y) \geq I_{th}$ represents a non-convex set while the objective function is still convex in regards to $p(y|x)$ and thus the entire problem in Eq.(5.2) would be a non-convex optimization.

As a direct result of Theorem III.1 , it could be deduced that any solution offered using techniques developed for solving convex optimization problems would be suboptimal. One such suboptimal solution was developed by [9] by reintroducing the original problem in the format of a Lagrange multiplier problem where the goal is to

$$min_{P_{Y|X}(y|x)} \quad I(X;Y) - \beta I(U;Y) - \sum_{x,y} \gamma(x) p(y|x) \tag{5.5}$$

where once again $\beta > 0$ is assumed to be a Lagrange multiplier connecting the two mutual information functions given to us by the user. Furthermore $\gamma(x)$ represents the constraints imposed by the probabilistic nature of the problem mainly how $\sum_{x,y} p(x,y) = 1$. Moreover, it is assumed that the normalization of the conditional probability matrix is imposed through the solution meaning I will definitely have $\sum_{y_i \in Y} P(y_i|x) = 1, \forall x \in \mathcal{X}$ and I need not worry about such series of conditions for the time being.

Before continuing to the derivation of the problem, I find it necessary to introduce another observation through Eq. (5.3) which was not previously mentioned in [9].

Theorem 5.3.2. *In the Lagrangian version of the problem as depicted in Eq.(5.3), I assume $\beta > 1$, otherwise the minimal value of the Lagrangian objective function will be equal to* 0.

The proof to this theorem is gathered in Appendix under Theorem III.2 in [59] and turns out to be quite straightforward.

As a direct result of Theorem III.2, from this point on, I will assume that $\beta > 1$ and focus on the answer to this class of problems.

5.3.2 An Analysis of the IB Method

In this subsection, I will offer an in-depth study of what [9] suggests in order to solve the problem formulated in Eq.(5.3).

In order to find the optimal $P_{Y|X}(y|x)$ to minimize $\mathcal{L} = I(X;Y) - \beta I(U;Y) - \sum_{x,y} \gamma(x)p(y|x)$, [9] opted to calculate the first order derivative of $\mathcal{L}$ with respect to $\mathbf{P}(\mathbf{y} = \mathbf{y}^*|\mathbf{x} = \mathbf{x}^*)$ for any possible x^* and y^* and have it be equal to 0. In other words, $P_{Y|X}(y|x)$ was chosen as the free variable.

To do so, [9] used the fact that

$$\begin{aligned} p(y) &= \sum_{x \in \mathbf{X}} p(x)p(y|x) \\ \rightarrow \frac{\partial p(y)}{\partial p(y|x)}|_{x=x^*, y=y^*} &= p(x^*) \end{aligned} \tag{5.6}$$

and the notion that

$$\begin{aligned} p(y|u) &= \sum_{x \in \mathbf{X}} p(x|u)p(y|x) \\ \rightarrow \frac{\partial p(y|u)}{\partial p(y|x)}|_{x=x^*, y=y^*} &= p(x^*|u) \end{aligned} \tag{5.7}$$

Having made the above assumptions, I would have

$$\frac{\partial \mathcal{L}}{\partial p(y|x)}|_{x=x^*, y=y^*} = 0 \rightarrow$$

$$\begin{aligned} p(x^*)\{\log p(y^*|x^*) + 1 + (\beta - 1)\log p(y^*) + (\beta - 1) \\ -\beta - \beta \sum_{u} p(u|x^*) \log \frac{p(y^*)p(u|y^*)}{p(u)}\} - \frac{\gamma(x^*)}{p(x^*)} = 0 \end{aligned} \tag{5.8}$$

It then follows that

$$\log p(y^*|x^*) = \log p(y^*) + \frac{\gamma(x^*)}{p(x^*)}$$
$$+\beta\{D_{KL}\{p(u|x^*)||p(u)\} - D_{KL}\{p(u|x^*)||p(u|y^*)\}\}$$
$$\rightarrow p(y^*|x^*) = p(y^*)$$
$$\times exp(-\beta\{D_{KL}\{p(u|x^*)||p(u|y^*)\}$$
$$-D_{KL}\{p(u|x^*)||p(u)\} - \frac{\gamma(x^*)}{\beta p(x^*)}\})$$
$$\rightarrow p(y^*|x^*) = p(y^*)exp\{-\beta Z(x^*, y^*)\} \tag{5.9}$$

where I have used the definition of KL-divergence as $D_{KL}\{\mathbf{P}(\mathbf{X})||\mathbf{Q}(\mathbf{X})\} = \sum_{x\in\mathbf{X}} P(x)\log\frac{P(x)}{Q(x)}$ [56].

Assuming that the original function $\mathcal{L}$ is convex (which I have shown and [9] has mentioned to not be true), then the result as gathered in Eq.(5.9) will help minimize $\mathcal{L}$.

However, there is still another issue which needs to be addressed. $p(y^*|x^*)$s as gathered in Eq.(5.9) do not necessarily indicate a set of probability variables; while they are always greater than 0, there is no guarantee that they will remain below 1 or even that the sum of all of its possible values follow the marginal property (both defined in Eq.(5.4)).

[9] suggested that in order to deal with this issue, I would redefine the optimal $p(y^*|x^*)$ as

$$p(y^*|x^*) = \frac{p(y^*)exp\{-\beta Z(x^*, y^*)\}}{\sum_{y^{**}} p(y^{**})exp\{-\beta Z(x^*, y^{**})\}} \tag{5.10}$$

By doing so, [9] has guaranteed that $\sum_{y\in Y} p(y|x) = 1$ and that $p(y|x) \leq 1$. However, normalizing $p(y^*|x^*)$ as seen in Eq.(5.10), takes away from the concept of an optimal $p(y^*|x^*)$ in the first place. Thus, even if convexity of $\mathcal{L}$ were guaranteed, by the above imposition, the calculated $p(y^*|x^*)$ is no longer the optimal minimizer

I was looking for. Later on, I will offer further details as to why such an argument is important and quite problematic.

We could thus conclude that in the calculation of the optimal $p(y^*|x^*)$ as described in [9] much has been left to desire.

In [9], once the optimal free variable $p(y^*|x^*)$ was calculated, it was assumed that I would treat $P_Y(y)$ as auxiliary variables and fix

$$p(y^*) = \sum_{x \in \mathbf{X}} p(x)p(y^*|x), \forall y^* \in \mathcal{Y} \tag{5.11}$$

Such an assumption would guarantee another series of conditions that all marginal probability distributions should satisfy. Finally, it was suggested that by repeatedly fixing $p(y^*)$ and then finding the optimal $p(y^*|x^*)$ and vice versa, I could reach a point of convergence where the overall function $\mathcal{L}$ is minimized.

We have already shown that for a fixed $p(y^*)$, $p(y^*|x^*)$s as formulated in Eq.(5.10) are not necessarily optimal (they are not moving in the minimization direction). In this section, I show that for a fixed set of $p(y^*|x^*)$s, $p(y^*)$s as formulated in Eq.(5.11) do not move in minimization direction either.

To show this, I assume two different functions $\mathcal{F} = E_{x,y}[\log \frac{P_{Y|X}(y|x)}{Q_Y(y)}]$ and $\mathcal{G} = E_{x,y}[\log \frac{P_{Y|X}(y|x)}{\sum_{x \in \mathbf{X}} P_X(x)P_{Y|X}(y|x)}]$.

Note that in the function of $\mathcal{F}$, $Q()$ is a legitimate probability measure on $\mathcal{Y}$, but not necessary satisfying Bayes rule as in Eq.(5.11). This is because I are attempting to show some relationship between $\mathcal{F}$ and $\mathcal{G}$ for a more general setting. It follows that

$$\begin{aligned}\mathcal{F} - \mathcal{G} &= \sum_{x,y} P_X(x)P_{Y|X}(y|x) \log \frac{\sum_x P_X(x)P_{Y|X}(y|x)}{Q_Y(y)} \\ &\geq \sum_{x,y} P_X(x)P_{Y|X}(y|x)\{1 - \frac{Q_Y(y)}{\sum_x P_X(x)P_{Y|X}}\} \\ &= 1 - 1 = 0\end{aligned} \tag{5.12}$$

where in the penultimate line I have used the log-inequality $\log x \geq 1 - \frac{1}{x}$ with $=$ only appearing when $x = 1$. (which in this scenario is equivalent to the condition that $Q_Y(y) = \sum_x P_X(x)P_{Y|X}, \forall y \in \mathcal{Y}$). It follows that if I define

$$\mathcal{K} = E_{x,y}[\log \frac{P_{Y|X}(y|x)}{Q_Y(y)}] - \beta E_{u,y}[\log \frac{P_{Y|U}(y|u)}{Q_Y(y)}] \tag{5.13}$$

and

$$\begin{aligned} \mathcal{M} = E_{x,y}[\log \frac{P_{Y|X}(y|x)}{\sum_{x \in \mathbf{X}} P_X(x)P_{Y|X}(y|x)}] \\ -\beta E_{u,y}[\log \frac{P_{Y|U}(y|u)}{\sum_{x \in \mathbf{X}} P_X(x)P_{Y|X}(y|x)}] \end{aligned} \tag{5.14}$$

Then, I will not be able to derive much about the difference between $\mathcal{K}$ and $\mathcal{M}$ as

$$\begin{aligned} \mathcal{K} - \mathcal{L} = \sum_{x,y} P_X(x)P_{Y|X}(y|x) \log \frac{\sum_x P_X(x)P_{Y|X}(y|x)}{Q_Y(y)} \\ -\beta \sum_{u,x,y} P_X(x)P_{U|X}(u|x)P_{Y|X}(y|x) \times \log \frac{\sum_x P_X(x)P_{Y|X}(y|x)}{Q_Y(y)} \end{aligned} \tag{5.15}$$

where the first sum is always positive and the second always negative. As a result, based on the value of β the overall difference may be either positive or negative. If I assume $\beta << 1$, then the overall difference will be positive and as a result fixing $Q_Y(y) = \sum_x P_X(x)P_{Y|X}(y|x)$ will minimize the overall difference. On the other hand, for $\beta >> 1$, the overall difference will be negative and as a result fixing $Q_Y(y) = \sum_x P_X(x)P_{Y|X}(y|x)$ will maximize the overall difference.

Overall, it could be deduced that fixing $Q_Y(y) = \sum_x P_X(x)P_{Y|X}(y|x)$ will not necessarily result in minimizing $\mathcal{L}$ for a fixed $P_{Y|X}(y|x)$ and is thus not optimal.

Overall, one of the primary contributions of [9] was to offer two closed-form formulas by whose iterations and many choices of starting variables, it could be shown that a final value of $\mathcal{L}$ could be reached. However, as was mentioned in [9], this overall value is not necessarily the minimum I was looking for. In this section, I went one step further and demonstrated how this non-optimality is not only due to the non-convexity of the original problem (as [9] had pointed out), but also due to the method [9] has offered.

By revisiting the original formulation of $\mathcal{L}$ it could be witnessed that for a fixed $I(X;Y)$, further increasing $I(U;Y)$ would result in the minimization of the overall $\mathcal{L}$.

As a result, I suggest treating $P_{Y|U}(y|u)$ as a completely independent set of variables. I will simply introduce a series of conditions required to be satisfied to guarantee the linear relationship between $P_Y(y)$, $P_{Y|X}(y|x)$ and $P_{Y|U}(y|u)$. Consequently, I propose to turn the objective function $\mathcal{L}$ as following:

$$\mathcal{L} = E_{x,y}[\log \frac{P_{Y|X}(y|x)}{Q_Y(y)}] - \beta E_{u,y}[\log \frac{P_{Y|U}(y|u)}{Q_Y(y)}] \tag{5.16}$$

Such an idea has 2 very important advantages in my optimization problem:

1: In Eq.(5.16), it could be seen that the overall $\mathcal{L}$ is a convex function of $P_{Y|X}(y|x)$ (check the proof for Theorem III.1) and thus minimization overall $P_{Y|X}(y|x)$ alone is calculable.

2: Assuming a fixed $P_{Y|X}(y|x)$ and $P_Y(y)$, I need to show that there exists a class of variables $P_{Y|U}(y|u)$ which will help minimize the overall $\mathcal{L}$. By utilizing the same idea introduced in [10] and assuming that $P_Y(y)$ and $P_{Y|X}(y|x)$ are fixed and introducing $\mathcal{F}' = E_{u,y}[\log \frac{Q_{Y|U}(y|u)}{P_Y(y)}]$ and $\mathcal{G}' = E_{u,y}[\log \frac{W_{Y|U}(y|u)}{P_Y(y)}]$ where $W_{Y|U}(y|u) = \frac{P_Y(y)P_{U|Y}(u|y)}{\sum_{y\in\mathbf{Y}} P_Y(y)P_{U|Y}(u|y)}$, I can write

$$\begin{aligned}\mathcal{G}' - \mathcal{F}' &= \sum_{u,y} P_Y(y)P_{U|Y}(u|y) \times \log \frac{P_Y(y)P_{U|Y}(u|y)}{Q_{Y|U}(y|u)\sum_{y^*} P_Y(y^*)P_{U|Y}(u|y^*)} \\ &\geq \sum_{u,y} P_{U|Y}(u|y)P_Y(y) - \sum_{u,y} Q_{Y|U}(y|u)P_U(u) \\ &= 1 - 1 = 0 \end{aligned} \tag{5.17}$$

where I have once again utilized the log inequality and = only holds true when

$$\begin{aligned} Q_{Y|U}(y|u) &= \frac{P_Y(y)P_{U|Y}(u|y)}{\sum_{y\in\mathbf{Y}} P_Y(y)P_{U|Y}(u|y)} \\ &= \frac{\sum_x P_{X|U}(x|u)P_{Y|X}(y|x)}{\sum_{x,y} P_{X|U}(x|u)P_{Y|X}(y|x)} \end{aligned} \tag{5.18}$$

where I have simplified the first line using some simple mathematical calculations. It follows that satisfying Eq.(5.18) results in maximizing $\mathcal{F}'$ and thus minimizing $\mathcal{L}$ for a fixed $P_{Y|X}(y|x)$ and $P_Y(y)$. Interestingly, this condition is a direct result of imposing an extension of the Bayes rules on variables $P_{Y|U}(y|u)$ and $P_{Y|X}(y|x)$. As a result, I choose to select a third class of variables namely $P_{Y|U}(y|u)$ as another set of auxiliary variables to be updated alternatively together with $P_Y(y)$, and $P_{Y|X}(y|x)$.

As was witnessed, the biggest fall-back of [9]'s suggestions appeared when an attempt was made to ensure that the Bayes conditions between $P(Y)$ and $P(Y|X)$ and that between $P(Y|U)$ and $P(Y|X)$ underlying the Markov Chain $U \to X \to Y$ are reinforced with respect to only one free set of variables $P(Y|X)$. In order to deal with these problems, I suggest the use of Augmented Lagrange Multiplier (ALM) Method and more specifically the Alternating Direction Method of Multipliers (ADMM) which I will discuss in the following section.

5.3.3 Introducing Penalty (cost) Functions

After careful observation of the issues introduced in the previous subsection, I could deduce that most of the deficiencies of [9] come from a restrictive manner of dealing with constrained non-convex optimization problems. In both calculations, I have sacrificed the optimal direction of reaching a solution for making certain a constraint is met at every step of the iteration. It would then make sense to try and find a solution which can satisfy an acceptable threshold of constraints at every step of the iteration while not diverging from the optimal path of reaching a solution. It turns out such a solution could be developed by introducing a series of cost functions.

5.3.4 Constrained Optimization using Augmented Lagrange Multipliers

As mentioned previously, the biggest issue with the IB solution lays in the strict nature of a constraint resulting in many inconsistencies while developing an iterative algorithm. More specifically, I either allow the constraints to be fully broken so as to develop a straightforward solution (the ideal $p(y^*|x^*)$ calculated in Eq.(5.9)) or I impose the constraints so rigidly that they result in the a misdirection in the solution as was the case for imposing the marginal property in Eq.(5.11).

A method of dealing with such scenarios was addressed by the Augmented Lagrange Multiplier method [12] which allows a level of flexibility from such constraints but also introduces a penalty function for such diversions. As a result, by controlling the value of the penalty coefficient, I will be able to both account for possible diversions and also keep them under a specific threshold. Furthermore, by imposing a large penalty function, I will be able to virtually find answers when the constraints are not broken at all.

A secondary source of problems in developing [9]'s method could be traced back to forming an iterative relationship between only 2 classes of variables (namely $P_{Y|X}(y|x)$ and $P_Y(y)$s. In [9], by simply fixing one of the classes, I would try and find the other optimal class of variables. Such a solution would impose further weight and constraints over one of the variable classes. For example in the problem at hand as exemplified in Eq.(5.3), if I fix $P_Y(y)$, the optimal value of $P_{Y|X}(y|x)$ needs to account for traces of a secondary class of variables tentatively known as $P_{Y|U}(y|u)$'s as demonstrated through Eq.(5.7). I could argue that by further introducing new classes of variables, I could cut then on such variable interconnections thus dealing with simpler problems. As a result, from this point on, I assume I would like to optimize $\mathcal{L}$ over 3 sets of variable classes $P_Y(y)$, $P_{Y|X}(y|x)$ and $P_{Y|U}(y|u)$.

To introduce the penalty coefficients in my constrained optimization problem, I first list all possible constraints in detail assuming I am searching over all sets of variables $P_Y(y), P(Y|X)$ and $P_{Y|U}(y|u)$s. I break these constraints into those of an inequality nature:

$$\begin{aligned} 0 \leq p(y|x) \leq 1, \forall x \in \mathcal{X}, \forall y \in \mathcal{Y} \\ 0 \leq p(y|u) \leq 1, \forall u \in \mathcal{U}, \forall y \in \mathcal{Y} \\ 0 \leq p(y) \leq 1, \forall y \in \mathcal{Y} \end{aligned} \tag{5.19}$$

and those of an equality nature such as:

$$\begin{gathered} \sum_x p(x)p(y|x) = p(y), \quad \forall y \in \mathcal{Y} \\ p(y|u) = \frac{\sum_x p(x|u)p(y|x)}{\sum_{x,y} p(x|u)p(y|x)}, \quad \forall y \in \mathcal{Y}, \forall u \in \mathcal{U} \\ \sum_y p(y|x) = 1, \quad \forall x \in \mathcal{X} \\ \sum_y p(y|u) = 1, \quad \forall u \in \mathcal{U}, \quad \sum_y p(y) = 1 \end{gathered} \tag{5.20}$$

Note: It should be noted that all the variables $p(y|u)$ and $p(y)$ are linear functions of the original variables $p(y|x)$ and thus, the problem is still that of an optimization over the matrix $P_{Y|X}(y|x)$ with all the constraints over the same matrix. However, in order to further simplify the calculations and use the results from other works specifically ADMM [60], I treat $p(y|u)$ and $p(y)$ as separate variables and then use the set of constraints gathered in Eq.(5.19) and Eq.(5.20) as new constraints within theses variables.

5.3.5 New Problem Formulation using Penalties

To offer the new formulation, I first need to introduce some definitions:

1. Any $\lambda, \lambda', \lambda'', \lambda''', \lambda''''$ represents the Lagrange multipliers defined over their corresponding set of equality constraints and their indices simply reveal over which variables they are defined.
2. Any μ, μ' represents the Lagrange multipliers defined over their corresponding set of inequality constraints and their indices simply reveal over which variables they are defined.

Using the concept of ALM over constrained sets, I could write the overall new utility function in the format of $\mathcal{L}_\mathbf{c}$ as described in [59].

I need to address two new definitions in this new formulation:

(1) There is a use of *max* function in the formulation of $\mathcal{L}_\mathbf{c}$ which is due to the fact that I have chosen to rewrite the inequality constraints in the format of equality constraints using a positive variable. I then find the optimal value for such positive variables to minimize the overall utility function. This then in turn results in choosing a value between the optimal and 0, then if the optimal is positive, the optimal is chosen and otherwise, 0 is chosen which is the optimal allowed value [12].

(2) I have chosen a cost scalar c which controls the level of allowed invasion of constraints over the course of optimization[12].

5.3.6 Solution Derivation

We find the optimal set of answers $P_{Y|X}(y|x)$, $P_Y(y)$ and $P_{Y|U}(y|u)$ over which the function $\mathcal{L}_\mathbf{c}$ is minimized. I could carry this out in 3 different ways:

(1) I could solve the problem using the original Augmented Lagrange Multiplier Method as described in [12]. To do so, I will need to carry out the gradient descent method over a total of $|Y|$ variables (to account for all $p(y)$s) plus $|X| \times |Y|$ variables (to account for all $p(y|x)$s) plus $|U| \times |Y|$ variables (to account for all $p(y|u)$s) at the

same time. Such work, while computationally doable, is (1) very time-consuming and (2) does not offer a high convergence rate. This limit in convergence is due to the fact that I am trying to minimize an objective function $\mathcal{L}$ over a large number of variables each of whom impose their own set of constraints (λ 's and μs) and all of which need to be at least partially satisfied.

(2) I could solve this problem by optimizing $\mathcal{L}$ over different variable classes separately. To do so, I assume all $P_{Y|X}(y|x)$ and $P_{Y|U}(y|u)$s are fixed and then try to minimize $\mathcal{L}$ over all possible $P_Y(y)$s. I then fix $P_Y(y)$ and $P_{Y|X}(y|x)$ and try to optimize $\mathcal{L}$ over all $P_{Y|U}(y|u)$s and so forth. If I continue to iteratively optimize $\mathcal{L}$ over such sets, I could then hope that I will end up at a desirable minimum value for $\mathcal{L}$. By doing so, I will be able to cut down on the level of complications arising from optimization over a large number of variables as was the case in the original Augmented Lagrange Multiplier Method (ALM) and thus multiply the rate of convergence. This method is widely referred to as Alternating Direction Method of Multipliers (ADMM) and has been used in many recent studies to discuss non-convex optimization problems (as is the case in my problem).

(3) I could opt to use randomized ADMM [61] which is simply an extension of the original ADMM method as described in (2) where at every step of optimization, the order of optimization over different sets of varibale classes $P_{Y|X}(y|x)$, $P_{Y|U}(y|u)$ and $P_Y(y)$ is randomly selected. In other words, at every step, 1 of the 6 possible permutations of these variables is randomly chosen and then ADMM is carried out. While never mathematically proven, this method has been shown to offer a better optimization result through many practical implementations. [61, 62]

Note: It is important to note that in both ALM and ADMM, optimization is carried out through gradient descent. The only difference is that by using ADMM, I am breaking down a large scale ALM problem to a number of smaller scale ALM problems thus raising my chances of a converging solution. I thus suffice to solving the original problem using ADMM.

As for the Lagrange multipliers, I choose to update them at every $t+1^{th}$ step as follows:

$$\begin{aligned}
\lambda_{y^*}{}^{(t+1)} &= \lambda_{y^*}^{(t)} + c\{p^{(t)}(y^*) - \sum_x p(x)p^{(t)}(y^*|x)\} \\
\lambda^{'(t+1)}_{u^*,y^*} &= \lambda\lambda^{'(t)}_{u^*,y^*} + c\{p^{(t)}(y^*|u^*) - \frac{\sum_x p(x|u)p(y|x)}{\sum_{x,y} p(x|u)p(y|x)}\} \\
\lambda^{''(t+1)} &= \lambda^{''(t)} + c\{-1 + \sum_y p^{(t)}(y)\} \\
\\
\lambda^{'''(t+1)}_{u^*} &= \lambda^{'''(t)}_{u^*} + c\{-1 + \sum_y p^{(t)}(y|u)\} \\
\lambda^{''''(t+1)}_{x^*} &= \lambda^{''''(t)}_{x^*} + c\{-1 + \sum_y p^{(t)}(y|x)\} \\
\mu^{(t+1)}_{x^*y^*} &= \frac{\mu^{(t)}_{x^*y^*}}{2}, \qquad \mu^{'(t+1)}_{x^*y^*} = \frac{\mu^{'(t)}_{x^*y^*}}{2} \\
\mu^{(t+1)}_{u^*y^*} &= \frac{\mu^{(t)}_{u^*y^*}}{2}, \qquad \mu^{'(t+1)}_{u^*y^*} = \frac{\mu^{'(t)}_{u^*y^*}}{2} \\
\mu^{(t+1)}_{y^*} &= \frac{\mu^{(t)}_{y^*}}{2}, \qquad \mu^{'(t+1)}_{y^*} = \frac{\mu^{'(t)}_{y^*}}{2}
\end{aligned} \tag{5.21}$$

where the overheads t and $t+1$ for a certain variable represent the value of said variable in t and $t+1$ steps. Finally, I follow these steps as an algorithm of solving the Information Bottleneck problem:

Step 0: Choose random initial values for $p^{(0)}(y^*|x^*)$,$p^{(0)}(y^*|u^*)$ and $p^{(0)}(y^*)$ and all $\lambda^{(0)}$s and $\mu^{(0)}$s.

Step 1: Run Gradient Descent Algorithm to find $p^{(t+1)}(y^*|x^*)$ which minimizes $\mathcal{L}_\mathbf{c}$ for fixed values of $p^{(t)}(y^*|u^*)$ and $p^{(t)}(y^*)$ and all $\lambda^{(t)}$s and $\mu^{(t)}$s.

Step 2: Run Gradient Descent Algorithm to find $p^{(t+1)}(y^*|u^*)$ which minimizes $\mathcal{L}_\mathbf{c}$ for fixed values of $p^{(t+1)}(y^*|x^*)$ and $p^{(t)}(y^*)$ and all $\lambda^{(t)}$s and $\mu^{(t)}$s.

Step 3: Run Gradient Descent Algorithm to find $p^{(t+1)}(y^*)$ which minimizes $\mathcal{L}_\mathbf{c}$ for fixed values of $p^{(t+1)}(y^*|x^*)$, $p^{(t+1)}(y^*|u^*)$ and all $\lambda^{(t)}$s and $\mu^{(t)}$s.

Step 4: Use Eq.(5.21) to update $\lambda^{(t+1)}$s and $\mu^{(t+1)}$s.

Step 5: Go to step 2 unless all variables from steps t and $t+1$ are equal.

5.4 Numerical Results and Comparison

In this section, I run 3 different algorithms upon 3 different settings ($U \to X$ probability distributions channels) assuming that in all cases $|X| = |U| = 3, |Y| = 2$. For all cases, I have assumed that

$$\mathbf{P}(\mathbf{X}|\mathbf{U}) = \begin{bmatrix} 0.3 & 0.1 & 0.6 \\ 0.8 & 0.1 & 0.1 \\ 0.5 & 0.4 & 0.1 \end{bmatrix} \tag{5.22}$$

and

$$\begin{aligned} (1)\mathbf{P}(\mathbf{U}) &= \begin{bmatrix} 0.3 & 0.4 & 0.3 \end{bmatrix} \\ (2)\mathbf{P}(\mathbf{U}) &= \begin{bmatrix} 0.4 & 0.4 & 0.2 \end{bmatrix} \end{aligned} \tag{5.23}$$

The algorithms I opt to use are the original IB method [9], my proposed ADMM based algorithm and a randomized ADMM method.

5.4.1 Grounds for Comparison

In order to fairly compare the algorithms, I take note of how the IB method requires a given constant $\beta > 1$.

As a result, my first measure of comparison is to plot the final calculated value of the objective function $\mathcal{L}$ versus differing values of $\beta > 1$. Then, one method of comparing the efficiency of each method is to see which one of them offers a minimal $\mathcal{L}$ and is thus the optimal in my scenario. By doing so, I are inadvertently choosing the probability distribution matrix which helps minimize the function gathered in Eq.(5.3). These results are gathered in part (2) of Figures 5.1 and 5.2 respectively.

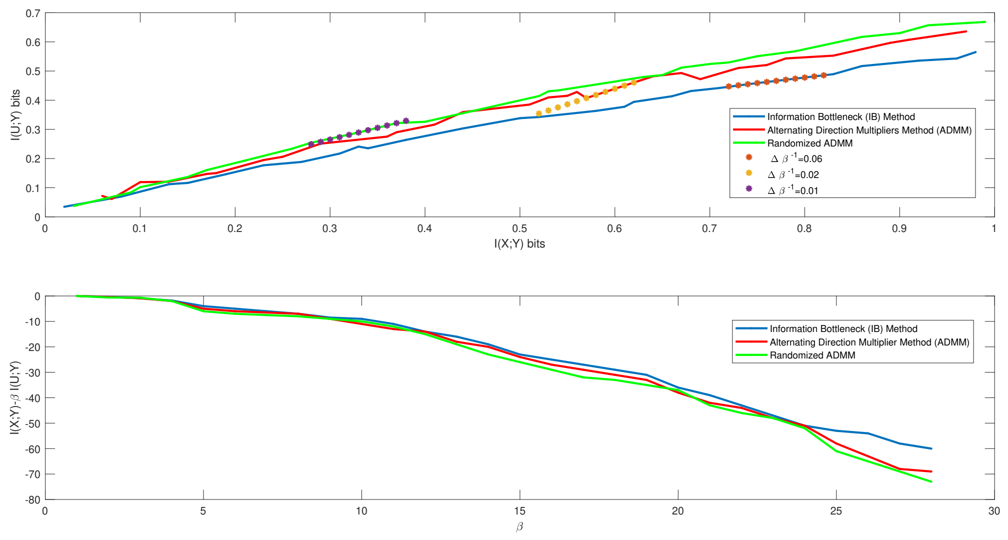

FIGURE 5.1. (1) $I(U;Y)$ for a fixed $I(X;Y)$ and (2) $\mathcal{L}$ for Example 1

Furthermore, I opt to compare different values of $I(U;Y)$ for a fixed value of $I(X;Y)$. By doing so, I am somehow moving away from the Lagrange multiplier nature of the problem and instead focusing on the more constrained optimization nature of it instead (as better formulated in Eq.(5.2)). Then, one method of comparing the efficiency of each method is to see which one of them offers the maximal $\mathcal{L}$ and is thus the optimal choice in my scenario. The results are gathered in part (1) of Figures 5.1 and 5.2 respectively.

5.4.2 Final Results

As can be witnessed, in both series of results, the optimal value is achieved through the use of randomized ADMM, followed by normal ADMM followed by the original IB method. As expected, there are times when two or even all 3 of the algorithms result in the same overall objective function however there are many times when one overpowers the other 2.

Finally, I would like to offer further insight as to why I have chosen to plot graphs representing $I(U;Y)$ vs $I(X;Y)$ at all. It might seems like just by plotting $\mathcal{L}$ for different values of $\beta > 1$, one might be able to judge the efficiency of randomized ADMM over all other methods. While that may be true, I am more interested in the relationship between these two variables.

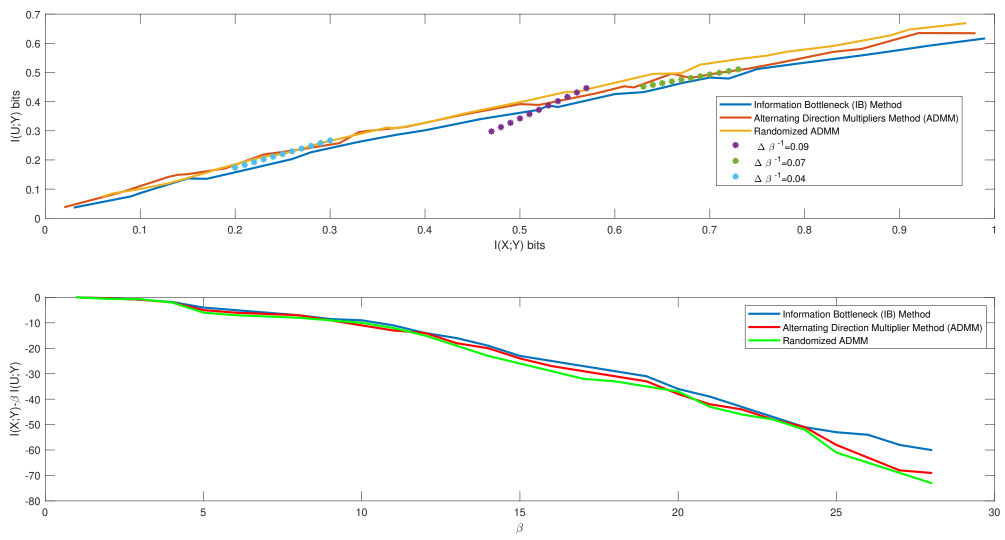

FIGURE 5.2. (1) $I(U;Y)$ for a fixed $I(X;Y)$ and (2) $\mathcal{L}$ for Example 2

[9] previously discussed how in an optimal setting (if the problem was convex), the optimal solution of the problem formulated in Eq.(5.3) would satisfy:

$$\frac{\partial I(U;Y)}{\partial I(X;Y)} = \frac{1}{\beta} \tag{5.24}$$

This is a direct conclusion from optimizing $\mathcal{L}$ by calculating its first derivative and putting it equal to 0. Using this intuition, I can now compare my solution with the optimal I am hoping to achieve for every possible solution.

We refer the reader to part (1) of Figures 5.1 and 5.2. Optimally speaking, I hope to find the part of the figure whose slope is the closest to $\frac{1}{\beta}$ for a given β. The slopes indicated on the figure represent the closest slopes I could get to $\frac{1}{\beta} = \frac{1}{2} = 0.5$. As can be seen, the least amount of difference between the two slopes $\Delta\frac{1}{\beta}$ belongs to randomized ADMM followed by normal ADMM followed by IB. This offers us another insight as to why randomized ADMM is the optimal solution to my problem at hand seeing as how it trails the optimal setting the closest.

5.4.3 Convergence versus Accuracy

As a means of better understanding how Augmented Lagrange Multiplier whether it be ALM or ADMM improves my chances of a solution, I decided to plot a series of figures indicating how different quality factors of the problem change based upon the value of the penalty coefficient (indicated as c during this chapter). The final results turned out to be quite interesting and well worth a deeper study. Parts (1)

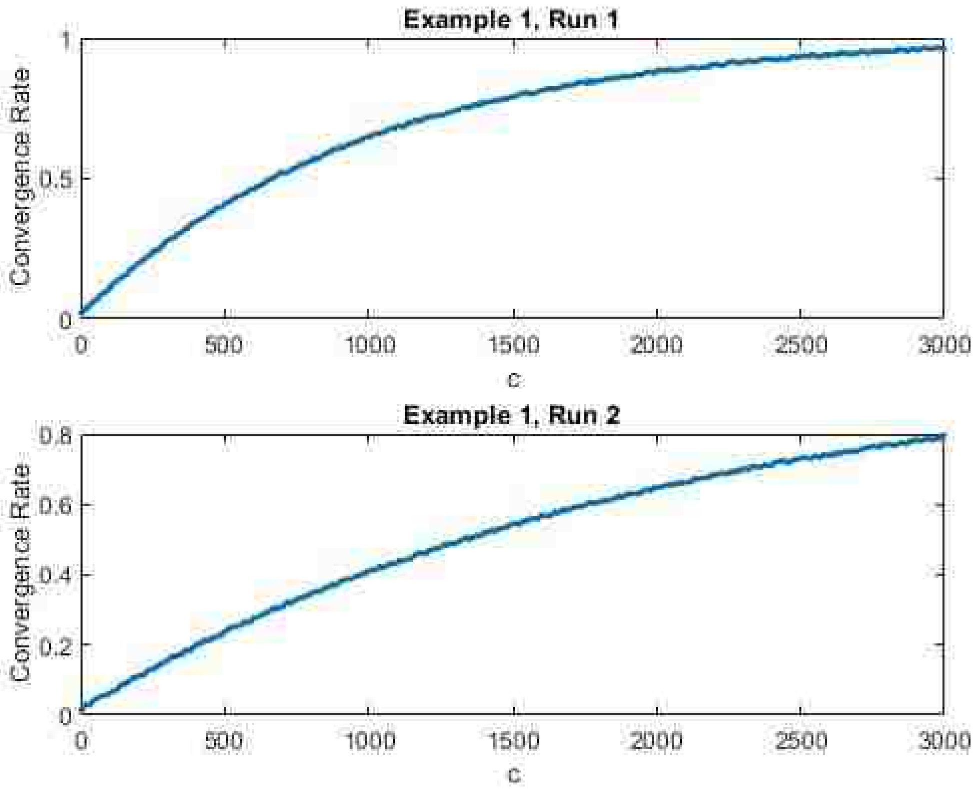

FIGURE 5.3. Convergence Rate vs c for Example 1 in (1) Run 1 (2) Run 2

and (2) of Figure 5.3 demonstrate the convergence rate of the ADMM solution versus the value of c for Example 1 in 2 runs of 10,000 repetitions. As can be seen, as a general rule, as c grows larger, the convergence rate grows higher and higher until for some large enough value of c, the method is almost always converging. Unfortunately, this large value of c is very case-dependent, as under the same circumstances and with simply different starting points (Step 0 of the algorithm) it could drastically change.

On the other hand, I have Figure 5.4 which demonstrates the accuracy of the solution I end up with at the end of running my algorithm. Such accuracy is calculated through the measure $\Delta\frac{1}{\beta}$ which was fully described previously. An accuracy of 1 means $\Delta\frac{1}{\beta} \to 0$ and an accuracy of almost zero refers to when $\Delta\frac{1}{\beta} \to max(\frac{1}{\beta}, 1-\frac{1}{\beta})$ where the 2 extremes of the max refer to when the closest slope to the desired β is infinity and 1 respectively. The resulting graph will be in the following format.

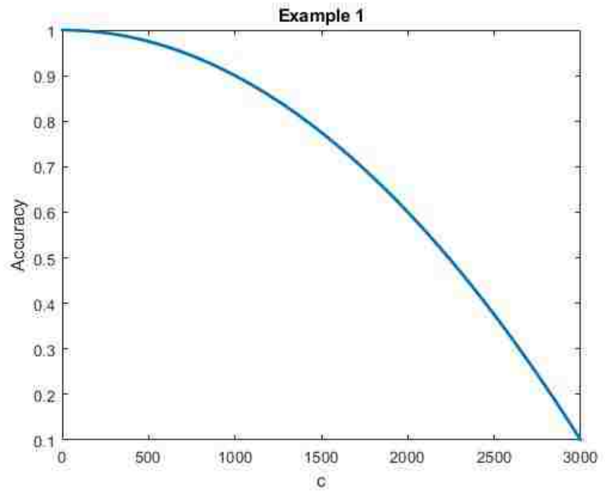

FIGURE 5.4. Accuracy Rate vs c for Example 1

It's quite interesting to see that the 2 measures of quality act completely opposite one another as c increases. In order to explain this phenomenon, I need to remind the reader of the meaning behind $\mathcal{L}_{\mathbf{c}}$. By allowing penalty coefficients in the form of quadratic functions, I am simplifying some of the more non-convex elements of the original objective function. It then follows that if c grows larger, the quadratic elements of the new objective function will become more prominent and could direct the entire objective function into a completely convex nature thus resulting in a very high convergence rate. However, if I keep increasing c with no regards to the original objective function, I will no longer be solving an optimization over $\mathcal{L}$ but rather a new objective mainly defined over c and thus while an answer will almost certainly exist (if c is large enough), it will start to diverge drastically from the desired optimization problem (a very low accuracy). [13]

Finally, I would like to investigate if an actual lower bound for c could be found to ensure that the overall objective function will converge. So far, in the 2 examples provided above, it could be witnessed that such a lower bound might not be truly calculable (as the threshold changes for the same problem settings with only different starting points).

[48] discusses a series of sufficient conditions whose satisfaction would result in the absolute convergence of a non-convex optimization problem. Namely, [48] introduces the concept of Lipschitz continuity over each select set of variable class and shows how if such a condition is granted, convergence is guaranteed. To do so, [48] introduces the concept of Lipschitz Differentiability which needs to hold true over functions of each separate variable set. Next, I will show that unfortunately such a series of conditions do not hold true in my scenario. All in all, it could be shown that such sufficient conditions do not hold true over my class of optimization problem and thus I could not use the results from [48] to make any further conclusions.

Chapter 6

Tradeoff between Disclosable and Private Latent Information revealed via Compressed One: an ADMM-Based Approach

6.1 Introduction

Previously in [49], I introduced mutual information between an input and an output as a measure of calculating the leaked information about the input sequences by an observer who only has access to the output sequences. More accurately, I discussed how mutual information is closely related to the number of input sequences decipherable by the observer assuming a long enough sequence of input has been produced and shared.

While the concept of leaked information [49] was initially introduced over a deterministic channel, as I will show, it could be further generalized to account for a channel of probabilistic nature as well. Using such notion, I could then reintroduce the original Information Bottleneck problem [9] described over the probabilistic channel $U \rightarrow X \rightarrow Y$ as an optimization problem where the goal is to minimize $I(X;Y)$ while keeping $I(U;Y)$ greater than a lower bound.

Using the same definition of the problem I can then introduce a new class of problems modeled as $(U_1, U_2) \rightarrow X \rightarrow Y$ where the goal is to minimize $I(X;Y)$ while keeping $I(U_1;Y)$ greater than a lower bound and $I(U_2;Y|U_1)$ less than a specific upper bound.

In such a model, U_1 and U_2 represent the disclosable and private aspects of the input respectively which may or may not be probabilistically intertwined. X represents a median variable available to the data owner which is the result of a probabilistic channel $\mathbf{P}(\mathbf{X}|\mathbf{U_1}, \mathbf{U_2})$ given (U_1, U_2) and Y represent a variable accessible to the observer. The goal of the observer is then to find a probabilistic channel between X and Y with as much compression which can reveal a lower bound threshold of information about U_1 while keeping the leaked information about U_2 given the revealed information about U_1 to an upper bound. The rest of this chapter will be dedicated to introducing methods of solving such a problem.

We recently became aware of the works done in [15] which closely resemble my overall formulation of the problem. However, in my work, my definition of the original problem and how I come to formulate it as such are quite different. Furthermore, even though information bottleneck alike approach has also been considered by us as [15], it should be noted this approach serves as a comparison reference, leading to my novel new algorithm. . Also of note is how [15] did not offer any numerical results for the method developed and simply sufficed to reiterate how the problem is non-convex (as was the original IB problem) and that the results are not necessarily optimal. While it is true that the overall problem is still non-convex, as I will show, there are methods that could outdo simply running the Information Bottleneck method again with little fixtures. I will detail such novel approaches and algorithms whose performances are compared with the IB based approach using numerical results.

In order to introduce my method, I find it necessary to introduce the concept of Augmented Lagrange Multiplier (ALM) [12] and later on Alternating Direction Multiplier Method (ADMM) [13]. Overall, this chapter is a further generalization of my previous work in [14] where I discussed applying ADMM to the original information bottleneck problem; however without an additional privacy constraint.

The rest of the chapter is organized as follows. In Section 6.2, I develop and formulate the problem at hand. In Section 6.3, I utilize the original Information Bottleneck method to develop an initial solution for the problem. I make note to discuss why this method might not be optimal. Then in Section 6.4, I develop 2 ADMM-based algorithms for solving the same problem. I proceed to indicate how and why each of these methods are well-worth studying. Afterwards, in Section 6.5, I offer many numerical results to showcase the superiority of my 2 methods over the original IB method and further compare the 2 of them in different regards. Finally in Section 6.6 I conclude the chapter.

6.2 System Model and Problem Formulation

We assume a probabilistic Markov chain $(U_1, U_2) \to X \to Y, x \in \mathcal{X}, y \in \mathcal{Y}$ where the initial probabilities of $u_1 \in \mathcal{U}_1$, $u_2 \in \mathcal{U}_2$ and the elements of the probabilistic matrix $\mathbf{P}(\mathbf{X}|\mathbf{U_1}, \mathbf{U_2})$ are given. It should further be noted that all random variables are assumed to be discrete with finite alphabet set. Then the goal of the problem would be to find the probabilistic matrix $\mathbf{P}(\mathbf{Y}|\mathbf{X})$ to attain the tradeoff between disclosable and Private Latent Information revealed via compressed One. To further elaborate upon this concept I deem it necessary to first introduce a measure of leaked information.

6.2.1 Revealed Information

In a memoryless probabilistic channel $V \to Z$ where the joint distribution $P(V, Z)$ is known, there is ambiguity and uncertainty when observing the output sequence about the specific input sequence over a successive n i.i.d visits. More specifically, if the input is a sequence of n independent realization of V symbols, the output sequence would be of form $\overrightarrow{\mathbf{V^n}} = [V_1, V_2, \cdots, V_n]$, out of $|V|^n$ possible outcomes. From an observer's perspective who can only have access to sequences appearing in the form of $\overrightarrow{\mathbf{Z^n}} = [Z_1 Z_2 ... Z_n]$, the goal is to classify the input sequence $\overline{\mathbf{V}}^n$ into a number of differential classes in the presence of uncertainty caused by the probabilistic mapping of the channel $\mathbf{P}(\mathbf{Z}|\mathbf{V}) = \prod_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{P}_{\mathbf{Z}|\mathbf{V}}(\mathbf{Z_i}|\mathbf{V_i})$ when n is sufficiently large. As a result, information about the specific input patterns is leaked to certain degree and can be measured using mutual information $I(V;Z)$ between V and Z, under a joint distribution $\mathbf{P}(\mathbf{V},\mathbf{Z})$. Such mutual information thus measures the maximum number of bits of meta information about item sequence per channel use. Therefore, I can have at most $2^{nI(V;Z)}$ sequences $\overline{\mathbf{V}^n}$ distinguishable by inferring based on $\overline{\mathbf{Z}^n}$ for large n [56]. I thus adopt $I(V;Z)$ as the privacy metric conditioned on a particular joint distribution $\mathbf{P}(\mathbf{V},\mathbf{Z})$.

Note that I am not considering a communication problem in which both transmitter and receiver share a codebook whose rate is below channel capacity so that the specific sequence representing a message from the sender can be restored without errors in asymptotic regime [56] . I do however rely on jointly typical sequences ideas to justify the operational meaning of $I(V;Z)$ to quantify the information disclosed about patterns of long input sequences.

6.2.2 Problem Formulation

Using this measure of leaked information, the problem will then be to find the optimal $\mathbf{P}(\mathbf{Y}|\mathbf{X})$ by which

Set of Requirements for $\mathbf{P}(\mathbf{Y}|\mathbf{X})$

- $I(X;Y)$ is minimized.
- $I(U_1;Y)$ is kept higher than a lower bound I_1.
- $I(U_2;Y|U_1)$ is kept lower than an upper bound I_2.

where $I(X;Y)$ represents the compression rate between X and Y; $I(U_1;Y)$ represents the revealed information about U_1 (public section of input) through Y and $I(U_2;Y|U_1)$ represents the revealed information about U_2 given the revealed information about U_1 through Y.

As a result, the overall problem could be formulated as:

$$\boxed{\begin{gathered}min_{p(y|x)}\mathcal{L},\\ \mathcal{L} = I(X;Y) - \beta I(U_1;Y) + \gamma I(U_2;Y|U_1),\\ \beta, \gamma > 0 \qquad (6.1)\end{gathered}}$$

In Sections 6.3 and 6.4 I will detail multiple methods of solving the problem formulated in Eq. (6.1).

Note: It can be shown that in order for Eq.(6.1) to be more meaningful, I must have $\beta > 1$. If I assume $\beta \leq 1$, then I will have:

$$\begin{aligned}\mathcal{L} &= I(X;Y) - \beta I(U_1;Y) + \gamma I(U_2;Y|U_1)\\ &\geq I(X;Y) - I(U_1;Y) + \gamma I(U_2;Y|U_1) \geq (1+\gamma)I(U_2;Y|U_1)\end{aligned} \tag{6.2}$$

Then, an optimal value could be found by making Y independent of X, and (U_1, U_2). Thus, throughout the rest of this chapter I assume $\beta > 1$.

Note: I find it necessary to offer two definitions that are going to repeatedly appear throughout this chapter; primary variables are the original variables over whom the optimization problem is carried out. Auxiliary variables on the other hand refer to variables defined throughout the chapter which are dependent on the primary variable. For example in Eq.(6.1), the variable $p(y|x)$ is a primary variable and the variable $p(y)$ is auxiliary seeing as how it could be calculated through a Bayes rule concerning $p(y|x)$.

6.3 Solving the Problem using the Information Bottleneck method

In this section I will formulate the problem in a mathematical format and find a solution for it using the information bottleneck method.

To do so, I rewrite the problem in the following Lagrange multiplier format:

$$min_{p(y|x)}I(X;Y) - \beta I(U_1;Y) + \gamma I(U_2;Y|U_1) + \sum_{x,y}\xi(x)p(y|x) \tag{6.3}$$

where $\xi(x)$ is chosen so as to guarantee the final $p(y|x)$ satisfies the Bayes rule that $p(y) = \sum_{x\in\mathcal{X}} p(x)p(y|x), \forall y \in \mathcal{Y}$.

The information bottleneck method states that a solution can be met if

The Idea behind Information Bottleneck Method

1. The first order derivative of the objective function with respect to the primary variable is put equal to 0 and the respective primary variable is found as a function of auxiliary variables.
2. The auxiliary variables are updated through the primary variable following the Bayes rule.
3. This process is done iteratively until a convergence criterion is met .

To further investigate how this method works for Eq. (6.1) I first need to find a series of derivatives in the following manner.

$$\begin{aligned} p(u_1,u_2,y) = p(u_1,u_2)p(y|u_1,u_2) = \\ p(u_1,u_2)\sum_x p(y|x)p(x|u_1,u_2) \\ \rightarrow \frac{\partial p(u_1,u_2,y)}{\partial p(y|x)}|_{y=y^*,x=x^*} = p(u_1,u_2,x^*) \end{aligned} \tag{6.4}$$

where $x*$ and y^* represent specific values of $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ wherein the derivative of $p(u_1,u_2,y)$ is calculated. Using the same logic I could deduce that

$$\begin{aligned} \frac{\partial p(u_1,y)}{\partial p(y|x)}|_{y=y^*,x=x^*} = p(u_1,x^*) \\ \frac{\partial p(y)}{\partial p(y|x)}|_{y=y^*,x=x^*} = p(x^*) \end{aligned} \tag{6.5}$$

We can now calculate the first order derivative with respect to $\mathbf{P}(\mathbf{Y}|\mathbf{X})$:

$$\frac{\partial \mathcal{L}}{\partial p(y|x)}|_{y=y^*,x=x^*} = 0 \rightarrow p(x^*)\log(\frac{p(y^*|x^*)}{p(y*)}) - \beta p(x^*)\times$$
$$\{-D_{KL}\{p(u_1|x^*)||p(u_1|y^*)\} + D_{KL}\{p(u_1|x^*)||p(u_1)\}\} + \gamma p(x^*)$$
$$\{\sum_{u_1} p(u_1|x^*)\{-D_{KL}\{p(u_2|u_1,x^*)||p(u_2|u_1,y^*)\} + D_{KL}\{p(u_2|u_1,x^*)||p(u_2)\}\}\}$$
$$+\xi(x^*) = 0 \tag{6.6}$$

where $D_{KL}[q_1(\omega)||q_2(\omega)]$ represents the KL-divergence between two probability distributions q_1, q_2 defined over a variable ω. Thus, I could deduce that a relationship in the manner of an exponential ratio for $\frac{p(y^*|x^*)}{p(y^*)}$ could be developed where:

$$p(y^*|x^*) = p(y^*)exp\{Z(x^*,y^*)\},$$
$$Z(x^*,y^*) = +\beta\{-D_{KL}\{p(u_1|x^*)||p(u_1|y^*)\}+$$
$$D_{KL}\{p(u_1|x^*)||p(u_1)\}\}$$
$$-\gamma\{\sum_{u_1} p(u_1|x^*)\{-D_{KL}\{p(u_2|u_1,x^*)||p(u_2|u_1,y^*)\}$$
$$+D_{KL}\{p(u_2|u_1,x^*)||p(u_2)\}\}\} + \frac{\xi(x^*)}{p(x^*)} \tag{6.7}$$

As can be witnessed in Eq.(6.7), I have managed to find the optimal $p(y^*|x^*)$ as a function of auxiliary variables $p(y^*), p(u_1|y^*)$ and $p(u_2|u_1,y^*)$. So, I need to offer a method of applying the Bayes rule over these 3 auxiliary variables. It follows that

$$p(y^*) = \sum_{x\in\mathcal{X}} p(x)p(y^*|x), \forall y^* \in \mathcal{Y} \tag{6.8}$$

$$p(u_1|y^*) = \frac{p(u_1)\sum_{x\in\mathcal{X}} p(x|u_1)p(y^*|x)}{p(y^*)} \tag{6.9}$$

$$p(u_2|u_1, y^*) = \frac{p(u_1, u_2) \sum_{x \in \mathcal{X}} p(x|u_1, u_2) p(y^*|x)}{p(y^*) p(u_1|y^*)} \tag{6.10}$$

We can thus develop the information bottleneck algorithm as gathered in Algorithm 1 where t and $t+1$ represent the iteration number or rather the iteration states. Thus by following Algorithm 1 and choosing enough random initial points for $p(0)(y|x)$, I am more than likely to find a point of convergence. However; (1) there is no guarantee that with very few random initial points $p(0)(y|x)$ I will converge (2) different initial points could lead to different convergence points many or all of which may not be optimal. (as I will further explain in Subsection 6.3.3)

It is important to note that the steps gathered in Eq.(6.8,6.9,6.10) represent the imposing of Bayes rule over the probabilistic channel as a whole. Such a solution was also developed in [15] where the original problem was defined in the format of fair supervision of data.

Algorithm 1: Optimizing $\mathcal{L}$ using IB

Step 0: Choose random initial values for $p^{(0)}(y^*|x^*)$ and use Eq. (6.8), (6.9) and (6.10) to find an initial value for the auxiliary variables $p^{(0)}(y^*), p^{(0)}(u_1|y^*)$ and $p^{(0)}(u_2|u_1, y^*)$.

Step 1: Use Eq. (6.7) to find $p^{(t+1)}(y^*|x^*)$ assuming all auxiliary variables are in their $(t)^{th}$ states.

Step 2: Use Eq. (6.8) to find $p^{(t+1)}(y^*)$ with the latest states of $p(y^*|x^*), p(u_1|y^*)$ and $p(u_2|u_1, y^*)$.

Step 3: Use Eq. (6.9) to find $p^{(t+1)}(u_1|y^*)$ with the latest states of $p(y^*|x^*), p(y^*)$ and $p(u_2|u_1, y^*)$.

Step 4: Use Eq. (6.10) to find $p^{(t+1)}(u_2|u_1, y^*)$ with the latest states of $p(y^*|x^*)$, $p(y^*)$ and $p(u_1|y^*)$.

Step 5: Go to step 2 unless all variables from step t and $t+1$ are equal.

However, as was explained in [14], imposing such Bayes rules are not necessarily pushing the overall value of $\mathcal{L}$ in a desired direction and might result in the overall value being non-optimal. I thus choose to first demonstrate these shortcomings and then follow my steps from [14] where I will explore the inherent structure of the objective function after introduction of a set of properly defined auxiliary variables to enable a more efficient and effective ADMM based algorithm than those offered by Information Bottleneck.

6.3.1 Optimal Probabilistic Channel for a Rate-Distortion Problem

It was proven [10] that the optimal channel mapping $X \to Y$ in a rate-distortion problem can be found in an iterative manner when the input is a given $p(x)$. In particular, the goal of minimizing $I(X;Y)$, for a fixed output distribution $p(y)$, can be achieved by finding the optimal $p(y|x)$ by solving a convex problem. Then, given a fixed channel mapping $p(y|x)$ the optimal $p(y)$ follows the Bayes rule in that $p(y) = \sum_x p(x)p(y|x)$.

To show this, I assume two different functions $\mathcal{F} = E_{x,y}[\log \frac{P_{Y|X}(y|x)}{Q_Y(y)}]$ and $\mathcal{G} = E_{x,y}[\log \frac{P_{Y|X}(y|x)}{\sum_{x\in\mathbf{X}} P_X(x)P_{Y|X}(y|x)}]$.

Note that in the function of $\mathcal{F}$, $Q()$ is a legitimate probability measure on $\mathcal{Y}$, but not necessary satisfying Bayes rule. This is because I am attempting to show some relationship between $\mathcal{F}$ and $\mathcal{G}$ for a more general setting. It follows that

$$\begin{aligned}\mathcal{F} - \mathcal{G} &= \sum_{x,y} P_X(x)P_{Y|X}(y|x)\log\frac{\sum_x P_X(x)P_{Y|X}(y|x)}{Q_Y(y)} \\ &\geq \sum_{x,y} P_X(x)P_{Y|X}(y|x)\{1 - \frac{Q_Y(y)}{\sum_x P_X(x)P_{Y|X}}\} \\ &= 1 - 1 = 0 \end{aligned} \tag{6.11}$$

where in the penultimate line I have used the log-inequality $\log x \geq 1 - \frac{1}{x}$ with $=$ only appearing when $x = 1$. (which in this scenario is equivalent to the condition that $Q_Y(y) = \sum_x P_X(x)P_{Y|X}, \forall y \in \mathcal{Y})$. Thus, it could be concluded that given a fixed channel mapping $p(y|x)$ the optimal $p(y)$ follows the Bayes rule in that $p(y) = \sum_x p(x)p(y|x)$.

6.3.2 Optimal Probabilistic Channel for a Channel Capacity Problem

Similar to the previous subsection, to solve the dual problem, i.e. channel capacity problem, under a given channel mapping $p(y|x)$, the goal of maximizing $I(X;Y)$ for a fixed posterior probability $p(x|y)$, can be achieved by finding the optimal $p(x)$. Then, given a fixed $p(x)$, the optimal $p(x|y)$ follows the Bayes rule in that $p(x|y) = \frac{p(x)p(y|x)}{\sum_x p(x)p(y|x)}$ [10].

To show this, I assume two different functions $\mathcal{F} = E_{x,y}[\log \frac{Q_{X|Y}(x|y)}{P_X(x)}]$ and $\mathcal{G} = E_{x,y}[\log \frac{\frac{P_X(x)P_{Y|X}(y|x)}{\sum_x P_X(x)P_{Y|X}(y|x)}}{P_X(x)}]$.

Note that in the function of $\mathcal{F}$, $Q()$ is a legitimate probability measure on $\mathcal{X}$, but not necessarily satisfying Bayes rule. This is because I am attempting to show some relationship between $\mathcal{F}$ and $\mathcal{G}$ for a more general setting. It follows that

$$\mathcal{G} - \mathcal{F} =$$

$$\sum_{x,y} P_X(x)P_{Y|X}(y|x) \log \frac{P_X(x)P_{Y|X}(y|x)}{Q_{X|Y}(x|y)\sum_x P_X(x)P_{Y|X}(y|x)}$$

$$\geq 1 - 1 = 0 \tag{6.12}$$

where in the penultimate line I have used the log-inequality $\log x \geq 1 - \frac{1}{x}$ with $=$ only appearing when $x = 1$. (which in this scenario is equivalent to the condition that $Q_{X|Y}(x|y) = \frac{P_X(x)P_{Y|X}(y|x)}{\sum_x P_X(x)P_{Y|X}(y|x)}, \forall x \in \mathcal{X}, y \in \mathcal{Y})$. Thus, it could be concluded that given a fixed channel mapping $p(y|x)$ the optimal $p(y)$ follows the Bayes rule in that $p(x|y) = \frac{p(x)p(y|x)}{\sum_x p(x)p(y|x)}$.

6.3.3 Results from Optimal Probabilistic Channel Settings

In this subsection, I utilize the results from Subsections 6.3.1,6.3.2 to derive some results for the implementation of IB on the problem formulated in Eq.(6.1).

We can see that

$$
\begin{aligned}
I(X;Y) &= E_{x,y}[\log \frac{p(y|x)}{p(y)}] = -E_{x,y}[\log \frac{p(y)}{p(y|x)}] \\
I(U_1;Y) &= E_{u_1,y}[\log \frac{p(y|u_1)}{p(y)}] = -E_{x,y}[\log \frac{p(y)}{p(y|u_1)}] \\
I(U_2;Y|U_1) &= E_{u_1,u_2,y}[\log \frac{p(y|u_1u_2)p(u_1u_2)}{p(u_1)p(u_2)p(y|u_1)}] - E_{u_1,u_2}[\log \frac{p(u_2|u_1)}{p(u_2)}] = \\
&-E_{u_1,u_2,y}[\log \frac{p(u_1)p(u_2)p(y|u_1)}{p(y|u_1u_2)p(u_1u_2)}] - E_{u_1,u_2}[\log \frac{p(u_2|u_1)}{p(u_2)}] \qquad (6.13)
\end{aligned}
$$

As a result, imposing Bayes rule on $p(y)$ based upon $p(y|x)$ (and $p(y|u_1,u_2)$ by Markov Chain) will result in the minimization of both $I(X;Y)$ and $I(U_1;Y)$ which is not necessarily desirable as I seek to maximize $I(U_1;Y)$. On the other hand, imposing Bayes rule on $p(y|u_1)$ based upon $p(y)$ (and $p(y|u_1,u_2)$ by Markov Chain) will result in the maximization of $I(U_1;Y)$ and minimization of $I(U_2;Y|U_1)$ which is completely desirable.

Rather, fixing the variables using Bayes rule is not necessarily an optimal option.

6.4 ADMM Solution

The nature of my proposed ADMM solution for the problem formulated in Eq.(6.1) is essentially to introduce cost functions for any violations of necessary Bayes rule (as was the case in Subsection 6.3.3) conditioned over the probabilistic channel $(U_1, U_2) \rightarrow X \rightarrow Y$ and then run separate gradient descent algorithms over different variables assuming all of them are primary variables.

6.4.1 Original ADMM

To develop a setting amenable to ADMM approach to the problem formulated in Eq.(6.1), I need to first introduce a series of primary and auxiliary variables over which I then find the set of necessary conditions that need to hold true to satisfy either the probabilistic nature of the variables or the Bayes rules imposed upon them. I then develop an ADMM version cost function of $\mathcal{L}$, namely, $\mathcal{L}'$ to be minimized instead.

Set of Constraints for Optimizing $\mathcal{L}^{'}$

$$p(y) = \sum_{x} p(x)p(y|x),\ p(y) = \sum_{u_1} p(u_1)p(y|u_1),\ \forall y \in \mathcal{Y}$$

$$p(y|u_1) = \sum_{u_2} p(u_2)p(y|u_1, u_2),\ \forall u_1 \in \mathcal{U}_1, y \in \mathcal{Y}$$

$$\sum_{y} p(y) = 1$$

$$\sum_{y} p(y|u_1) = 1 \quad \forall u_1 \in \mathcal{U}_1$$

$$\sum_{y} p(y|x) = 1, \quad \forall x \in \mathcal{X}$$

$$\sum_{y} p(y|u_1, u_2) = 1 \quad \forall u_1 \in \mathcal{U}_1, u_2 \in \mathcal{U}_2 \tag{6.14}$$

$$0 \leq p(y) \leq 1 \quad \forall y \in \mathcal{Y}$$

$$0 \leq p(y|x) \leq 1 \quad \forall x \in \mathcal{X}, y \in \mathcal{Y}$$

$$0 \leq p(y|u_1) \leq 1 \quad \forall u_1 \in \mathcal{U}_1$$

$$0 \leq p(y|u_1, u_2) \leq 1 \quad \forall u_1 \in \mathcal{U}_1, u_2 \in \mathcal{U}_2, y \in \mathcal{Y} \tag{6.15}$$

To do so, I fix all the primary variables except one and then find the optimal variable class to minimize the overall $\mathcal{L}^{'}$ repeatedly. In other words, I am taking the original Information Bottleneck solution (where the only goal was to optimize over the primary variable class and fix all auxiliary variables through it) and reintroduce the auxiliary variables as primary variables themselves and optimize them independently after making desirable changes to the original objective function $\mathcal{L}$.

To run the original version of ADMM over the problem formulated in Eq.(6.1), I simply assume 4 primary variable classes $p(y), p(y|x), p(y|u_1)$ and $p(y|u_1, u_2)$. These variables were chosen because they are the ones that originally came up in the formulation of $\mathcal{L}$ in Eq.(6.13). Over these classes I find the necessary set of conditions that need to hold true.

We then develop a new version of $\mathcal{L}$ named $\mathcal{L}'$ which is a costly version of $\mathcal{L}$ and attempt to minimize it instead. This costly version uses the concept of Augmented Lagrange Multipliers (ALM) [12] to include the constraints of an optimization problem in the objective function by adding a quadratic cost function to the overall objective function. Thus, any divergence from the assumed constraints will result in an overall larger objective function.

Algorithm 1 offers the method to solving such problem.

$$
\begin{aligned}
\mathcal{L}' = &\sum_{x,y} p(y|x) \log p(y|x) - \sum_{y} p(y) \log p(y) \\
&-\beta\{\sum_{u_1,y} p(y|u_1) \log p(y|u_1) - \sum_{y} p(y) \log p(y)\} \\
&+\gamma\{\sum_{u_1,u_2,y} p(u_1,u_2) p(y|u_1,u_2) \log p(y|u_1,u_2) \\
&\quad - \sum_{u_1,y} p(u_1) p(y|u_1) \log p(y|u_1)\} \\
&\quad + \sum_{i=1}^{7} EC_i + \sum_{j=1}^{8} IC_j
\end{aligned}
\tag{6.16}
$$

where

$$
EC_1 = \sum_{y} \lambda_y \{p(y) - \sum_{x} p(x) p(y|x)\} + \frac{c}{2} \sum_{y} \{p(y) - \sum_{x} p(x) p(y|x)\}^2
$$

$$
EC_2 = \sum_{y} \lambda'_y \{p(y) - \sum_{u_1} p(y) p(y|u_1)\} + \frac{c}{2} \sum_{y} \{p(y) - \sum_{u_1} p(u_1) p(y|u_1)\}^2
$$

$$EC_3 = \sum_{u_1,y} \lambda''_{u_1,y}\{p(y|u_1) - \sum_{u_2} p(u_2)p(y|u_1,u_2)\} + \frac{c}{2}\sum_{u_1,y}\{p(y|u_1) - \sum_{u_2} p(u_2)p(y|u_1,u_2)\}^2$$

$$EC_4 = \lambda'''\{\sum_y p(y) - 1\} + \frac{c}{2}\{\sum_y p(y) - 1\}^2$$

$$EC_5 = \sum_{u_1} \lambda''''_{u_1}\{\sum_y p(y|u_1) - 1\} + \frac{c}{2}\sum_{u_1}\{\sum_y p(y|u_1) - 1\}^2$$

$$EC_6 = \sum_{x} \lambda'''''_{x}\{\sum_y p(y|x) - 1\} + \frac{c}{2}\sum_{x}\{\sum_y p(y|x) - 1\}^2$$

$$EC_7 = +\sum_{u_1,u_2} \lambda''''''_{u_1,u_2}\{\sum_y p(y|u_1,u_2) - 1\} + \frac{c}{2}\sum_{u_1,u_2}\{\sum_y p(y|u_1,u_2) - 1\}^2 \quad (6.17)$$

represent the ALM version of the constraints gathered in Eq.(6.14) and

$$IC_1 = \frac{1}{2c}\sum_{y}\{max(0, \mu_y - cp(y))^2 - \mu_y^2\}$$

$$IC_2 = \frac{1}{2c}\sum_{x,y}\{max(0, \mu_{x,y} - cp(y|x))^2 - \mu_{x,y}^2\}$$

$$IC_3 = \frac{1}{2c}\sum_{u_1,y}\{max(0, \mu_{u_1,y} - cp(y|u_1))^2 - \mu_{u_1,y}^2\}$$

$$IC_4 = \frac{1}{2c}\sum_{u_1,u_2,y}\{max(0, \mu_{u_1,u_2,y} - cp(y|u_1,u_2))^2 - \mu_{u_1,u_2,y}^2\}$$

$$IC_5 = \frac{1}{2c}\sum_{y}\{max(0, \mu'_y - c + cp(y))^2 - \mu_y'^2\}$$

$$IC_6 = \frac{1}{2c}\sum_{x,y}\{max(0, \mu'_{x,y} - c + cp(x,y))^2 - \mu_{x,y}'^2\}$$

$$IC_7 = \frac{1}{2c}\sum_{u_1,y}\{max(0, \mu'_{u_1,y} - c + cp(y|u_1))^2 - \mu_{u_1,y}'^2\}$$

$$IC_8 = \frac{1}{2c}\sum_{u_1,u_2,y}\{max(0, \mu'_y - c + cp(y|u_1,u_2))^2 - \mu_{u_1,u_2,y}'^2\} \quad (6.18)$$

represent the ALM version of the constraints gathered in Eq.(6.15). Furthermore, λs, μs and c represent the equality constraint coefficients, inequality constraint coefficients and the cost coefficient respectively.

While running the simulation, I opt to run a randomized version of Algorithm 2 where at every iteration one of the $4! = 24$ permutations of steps 1,2,3 and 4 are chosen and then run in order. I will refer to this version of the algorithm as Randomized ADMM. (R-ADMM) [61]

```
**Algorithm 1: Minimizing $\mathcal{L}'$ using ADMM**
Step 0: Choose random initial values for $p^{(0)}(y|x), p^{(0)}(y), p^{(0)}(y|u_1), p^{(0)}(y|u_1, u_2)$ and all $\lambda^{(0)}$s and $\mu^{(0)}$s.
Step 1: Run Gradient Descent Algorithm to find $p^{(t+1)}(y|x)$ which minimizes $\mathcal{L}'$ for fixed values of $p^{(t)}(y), p^{(t)}(y|u_1), p^{(t)}(y|u_1, u_2)$ and all $\lambda^{(t)}$s and $\mu^{(t)}$s.
Step 2: Run Gradient Descent Algorithm to find $p^{(t+1)}(y)$ which minimizes $\mathcal{L}'$ for fixed values of $p^{(t+1)}(y|x), p^{(t)}(y|u_1), p^{(t)}(y|u_1, u_2)$ and all $\lambda^{(t)}$s and $\mu^{(t)}$s.
Step 3: Run Gradient Descent Algorithm to find $p^{(t+1)}(y|u_1)$ which minimizes $\mathcal{L}'$ for fixed values of $p^{(t+1)}(y|x), p^{(t+1)}(y), p^{(t)}(y|u_1, u_2)$ and all $\lambda^{(t)}$s and $\mu^{(t)}$s.
Step 4: Run Gradient Descent Algorithm to find $p^{(t+1)}(y|u_1, u_2)$ which minimizes $\mathcal{L}'$ for fixed values of $p^{(t+1)}(y|x), p^{(t+1)}(y), p^{(t+1)}(y|u_1)$ and all $\lambda^{(t)}$s and $\mu^{(t)}$s.
Step 5: Update $\lambda^{(t+1)}$s and $\mu^{(t+1)}$s.
Step 6: Go to step 2 unless all variables from steps $t$ and $t + 1$ are equal.
```

By this method, I am still imposing some of the Bayes rules I deem necessary while allowing some room for violations controlled by the cost variable c. As c grows larger, the set of necessary conditions become more restrictive (imposed) and if c becomes too small, none of the necessary conditions are assumed important.

6.4.2 Observant Randomized ADMM

By revisiting Eq.(6.13), it could be seen that the major drawback in utilizing Bayes rules for an optimal objective function was the fact that the same variables in different sections of the objective function (different mutual informations) reacted differently to imposing the Bayes rules over them. I thus find it interesting to see how the entire process would have worked if the same variables in different mutual information functions were treated as if they were independent of one another. We thus decide to break the original $\mathcal{L}$ into 3 different parts $I(X;Y), I(U_1;Y)$ and $I(U_2;Y|U_1)$ and optimize each of them using 2 independent sets of variables namely $p(y|x), p_1(y)$ for $I(X;Y)$; $p_1(y|u_1), p_2(y)$ for $I(U_1;Y)$ and finally $p_2(y|u_1), p(y|u_1,u_2)$ for $I(U_2;Y|U_1)$. For the time being, I assume that the variables with the same letter denotation and different indexes represent independent variables and introduce a cost function to reflect the cost of violation of their equality.

We now seek to see if I can simplify any of the set of constraints using the nature of each part of $\mathcal{L}$. To do so, I simply look at Eq.(6.13). Assuming that the same variables in different mutual information functions are independent, it could be deduced that the optimal value of $\mathcal{L}$ can only be achieved when $I(X;Y)$ and $I(U_2;Y|U_1)$ are minimized and $I(U_1;Y)$ is maximized. [10] has already shown that $I(X;Y)$ can be minimized when (1) the optimal $p(y|x)$ is found using gradient descent and (2) $p_1(y)$ is forced to follow the Bayes rule imposed by $p(y|x)$. By simply referring to Eq.(6.13) and noticing the similarity of the function to $I(X;Y)$, it could also be deduced that $I(U_2;Y|U_1)$ can be minimized when (1) the optimal $p(y|u_1,u_2)$ is found using gradient descent and (2) $p_2(y|u_1)$ is forced to follow the Bayes rule imposed by $p(y|u_1,u_2)$. Finally, through the same notation $I(U_1;Y)$ is maximized or rather $-I(U_1;Y)$ is minimized when (1) the optimal $p_1(y)$ is found using gradient descent and (2) $p_1(y|u_1)$ is forced to follow the Bayes rule imposed by $p_1(y)$.

Similarly, to solve the dual problem, i.e. channel capacity problem, under a given channel mapping $p(y|x)$, the goal of maximizing $I(X;Y)$ for a fixed posterior probability $p(x|y)$, can be achieved by finding the optimal $p(x)$. Then, given a fixed $p(x)$, the optimal $p(x|y)$ follows the Bayes rule in that $p(x|y) = \frac{p(x)p(y|x)}{\sum_x p(x)p(y|x)}$.

Using this notion, I can optimize each of the mutual functions separately by running one gradient descent to find an optimal variable and then fixing the other variable through the first one. Furthermore, I can cut down on some of the constraints I need to make certain are satisfied. Assuming the set of variables $(p(y|x), p_1(y), p_1(y|u_1), p_2(y), p_2(y|u_1), p(y|u_))$, the set of optimal constraints to impose in Observant ADMM (O-ADMM) can be developed as in Eq.(6.19) and Eq.(6.20).

Optimal Constraints to Impose in Observant ADMM (O-ADMM)

- For a fixed $p(y|x)$ and $I(U_1;Y)$ and $I(U_2;Y|U_1)$, the optimal value of $p_1(y)$ follows the Bayes rule and is equal to $p_1(y) = \sum_x p(x)p(y|x)$.

- For a fixed $p_2(y)$ and $I(X;Y)$ and $I(U_2;Y|U_1)$, the optimal value of $p_1(y|u_1)$ follows the Bayes rule and is equal to $p_1(y|u_1) = \frac{p(u_1|y)p_2(y)}{\sum_y p(u_1|y)p_2(y)}$.

- For a fixed $p(y|u_1,u_2)$ and $I(U_1;Y)$ and $I(X;Y)$, the optimal value of $p_2(y|u_1)$ follows the Bayes rule and is $p_2(y|u_1) = \sum_{u_2} p(u_2)p(y|u_1,u_2)$.

Set of Constraints for Optimizing $\mathcal{L}$

$$p_1(y) = p_2(y), \quad \forall y \in \mathcal{Y}$$

$$p_1(y|u_1) = p_2(y|u_1) \quad \forall u_2 \in \mathcal{U}_2, y \in \mathcal{Y}$$

$$\sum_y p_1(y) = 1$$

$$\sum_y p_1(y|u_1) = 1 \quad \forall u_1 \in \mathcal{U}_1$$

$$\sum_y p(y|x) = 1 \quad \forall x \in \mathcal{X}$$

$$\sum_y p(y|u_1, u_2) = 1 \quad \forall u_1 \in \mathcal{U}_1, u_2 \in \mathcal{U}_2 \tag{6.19}$$

$$0 \leq p_1(y) \leq 1, \quad \forall y \in \mathcal{Y}$$

$$0 \leq p(y|x) \leq 1 \quad \forall x \in \mathcal{X}, y \in \mathcal{Y}$$

$$0 \leq p_1(y|u_1) \leq 1 \quad \forall u_1 \in \mathcal{U}_1, y \in \mathcal{Y}$$

$$0 \leq p(y|u_1, u_2) \leq 1 \quad \forall u_1 \in \mathcal{U}_1, u_2 \in \mathcal{U}_\in, y \in \mathcal{Y} \tag{6.20}$$

We can then follow with the new objective function $\mathcal{L}''$ (which I have once more developed using Augmented Lagrange Multiplier method) and run the following algorithm:

$$\begin{aligned}\mathcal{L}'' = \sum_{x,y} p(y|x) \log p(y|x) - \sum_y p_1(y) \log p_1(y) \\ -\beta\{\sum_{u_1,y} p_1(y|u_1) \log p_1(y|u_1) - \sum_y p_2(y) \log p_2(y)\} \\ +\gamma\{\sum_{u_1,u_2,y} p(u_1,u_2) p(y|u_1,u_2) \log p(y|u_1,u_2) \\ -\sum_{u_1,y} p(u_1) p_2(y|u_1) \log p_2(y|u_1)\} \\ +\sum_{i=1}^{6} EC_i + \sum_{j=1}^{8} IC_j\end{aligned} \tag{6.21}$$

where ECs represent the costly representation of the equality constraints gathered in Eq.(6.19) and are calculated to be

$$EC_1 = \sum_y \lambda_y\{p_1(y) - p_2(y)\} + \frac{c}{2}\sum_y \{p_1(y) - p_2(y)\}^2$$
$$EC_2 = \sum_{u_1,y} \lambda'_{u_1,y}\{p_1(y|u_1) - p_2(y|u_1)\} + \frac{c}{2}\sum_{u_1,y}\{p_1(y|u_1) - p_2(y|u_1)\}^2$$
$$EC_3 = \lambda''\{\sum_y p_1(y) - 1\} + \frac{c}{2}\{\sum_y p_1(y) - 1\}^2$$
$$EC_4 = \sum_{u_1} \lambda'''_{u_1}\{\sum_y p_1(y|u_1) - 1\} + \frac{c}{2}\sum_{u_1}\{\sum_y p_1(y|u_1) - 1\}^2$$
$$EC_5 = \sum_x \lambda''''_x\{\sum_y p(y|x) - 1\} + \frac{c}{2}\sum_x\{\sum_y p(y|x) - 1\}^2$$
$$EC_6 = + \sum_{u_1,u_2} \lambda'''''_{u_1,u_2}\{\sum_y p(y|u_1,u_2) - 1\} + \frac{c}{2}\sum_{u_1,u_2}\{\sum_y p(y|u_1,u_2) - 1\}^2 \quad (6.22)$$

and ICs represent the costly representation of the inequality constraints gathered in Eq.(6.20) and are calculated to be

$$IC_1 = \frac{1}{2c}\sum_y \{max(0, \mu_y - cp_1(y))^2 - \mu_y^2\}$$
$$IC_2 = \frac{1}{2c}\sum_{x,y} \{max(0, \mu_{x,y} - cp(y|x))^2 - \mu_{x,y}^2\}$$
$$IC_3 = \frac{1}{2c}\sum_{u_1,y} \{max(0, \mu_{u_1,y} - cp_1(y|u_1))^2 - \mu_{u_1,y}^2\}$$
$$IC_4 = \frac{1}{2c}\sum_{u_1,u_2,y} \{max(0, \mu_{u_1,u_2,y} - cp(y|u_1,u_2))^2 - \mu_{u_1,u_2,y}^2\} \quad (6.23)$$

Algorithm 3: Minimizing $\mathcal{L}''$ using ADMM

Step 0: Choose random initial values for $p^{(0)}(y|x), p_1^{(0)}(y), p_1^{(0)}(y|u_1), p_2^{(0)}(y)$ and $p_2^{(0)}(y|u_1), p^{(0)}(y|u_1,u_2)$ and all $\lambda^{(0)}$s and $\mu^{(0)}$s.

Step 1: Run Gradient Descent Algorithm to find $p^{(t+1)}(y|x)$ which minimizes $\mathcal{L}'''$ for fixed values of $p_1^{(t)}(y), p_1^{(t)}(y|u_1), p_2^{(t)}(y), p_2^{(t)}(y|u_1), p^{(t)}(y|u_1,u_2)$ and all $\lambda^{(t)}$s and $\mu^{(t)}$s. Then impose $p_1^{(t+1)}(y) = \sum_x p(x)p^{(t+1)}(y|x)$.

Step 2: Run Gradient Descent Algorithm to find $p_2^{(t+1)}(y)$ which minimizes $\mathcal{L}'''$ for fixed values of $p^{(t+1)}(y|x), p_1^{(t+1)}(y), p_1^{(t)}(y|u_1), p_2^{(t)}(y|u_1), p^{(t)}(y|u_1,u_2)$ and all $\lambda^{(t)}$s and $\mu^{(t)}$s. Then impose $p_1^{(t+1)}(y|u_1) = \frac{p^{(t)}(u_1|y)p_2^{(t+1)}(y)}{\sum_y p^{(t)}(u_1|y)p_2^{(t+1)}(y)}$.

Step 3: Run Gradient Descent Algorithm to find $p^{(t+1)}(y|u_1,u_2)$ which minimizes $\mathcal{L}'''$ for fixed values of $p^{(t+1)}(y|x), p_1^{(t+1)}(y), p_2^{(t+1)}(y), p_1^{(t+1)}(y|u_1), p_2^{(t)}(y|u_1)$ and all $\lambda^{(t)}$s and $\mu^{(t)}$s. Then impose $p_2^{(t+1)}(y|u_1) = \sum_{u_2} p(u_2)p^{(t+1)}(y|u_1,u_2)$

Step 4: Update $\lambda^{(t+1)}$s and $\mu^{(t+1)}$s as done in [12].

Step 5: Go to step 2 unless all variables from steps t and $t+1$ are equal.

$$
\begin{aligned}
IC_5 &= \frac{1}{2c}\sum_{y}\{max(0,\mu'_y - c + cp_1(y))^2 - \mu'^2_y\}\\
IC_6 &= \frac{1}{2c}\sum_{x,y}\{max(0,\mu'_{x,y} - c + cp(x,y))^2 - \mu'^2_{x,y}\}\\
IC_7 &= \frac{1}{2c}\sum_{u_1,y}\{max(0,\mu'_{u_1,y} - c + cp_1(y|u_1))^2 - \mu'^2_{u_1,y}\}\\
IC_8 &= \frac{1}{2c}\sum_{u_1,u_2,y}\{max(0,\mu'_y - c + cp(y|u_1,u_2))^2 - \mu'^2_{u_1,u_2,y}\} \qquad (6.24)
\end{aligned}
$$

where λs, μs and c represent the equality constraint, inequality constraint and cost coefficients respectively.

While running the simulation, I opt to run a randomized version of Algorithm 3 where at every iteration one of the $3! = 6$ permutations of steps 1,2 and 3 are chosen and then run in order. Works such as [13] have shown that such randomization will help achieve better results. I will refer to this version of the algorithm as Observant Randomized ADMM. (OR-ADMM)

6.5 Numerical Results and Comparison

In this section I offer the numerical results of running any of the 3 methods, namely Information Bottleneck, Randomized ADMM, Observant Randomized ADMM.

However to do so, I first need to offer a measure of comparison between these methods:

6.5.1 Objective Function $\mathcal{L}$

The most straightforward measure of comparison between different methods would be to compare the original objective function $\mathcal{L} = I(X;Y) - \beta I(U_1;Y) + \gamma I(U_2;Y|U_1)$ for different values of β and γ. The method offering the lowest objective function will then be the optimal.

6.5.2 Non-optimality Measure

It should be noted that the overall goal of the original problem was to minimize $\mathcal{L}$ with an optimal selection of $p(y|x)$s. Thus, if a final value for $p(y|x)$s were found (any algorithm had converged), it would be optimal if and only if

$$\begin{gathered}\frac{d\mathcal{L}}{dp(y|x)} = 0 \rightarrow \\ \frac{dI(X;Y)}{dp(y|x)} - \beta\frac{dI(U_1;Y)}{dp(y|x)} + \gamma\frac{dI(U_2;Y|U_1)}{dp(y|x)} = 0 \\ \beta\frac{dI(U_1;Y)}{dI(X;Y)} - \gamma\frac{dI(U_2;Y|U_1)}{dI(X;Y)} - 1 = 0 \end{gathered} \tag{6.25}$$

We can thus introduce a measure of non-optimality which indicates how far the final solution of an algorithm is from the optimal solution. I choose

$$W = \beta\frac{dI(U_1;Y)}{dI(X;Y)} - \gamma\frac{dI(U_2;Y|U_1)}{dI(X;Y)} - 1 \tag{6.26}$$

and calculate it for different methods. The method with the closest absolute value of W to 0 will then be chosen as the optimal.

Seeing as how the original problem is non-convex, just running one iteration of the algorithm seems barely fair and I thus choose to run many examples of the algorithm and then make a judgement call over the entire results.

Furthermore, due to the discrete nature of the final values for $I(X;Y), I(U_1;Y)$ and $I(U_2;Y|U_1)$, (since they are all realized through runs of algorithms and thus not all their values exist) I use linear interpolations to calculate the derivative required in Eq.(6.26).

6.5.3 Accuracy and Convergence versus cost

Another measure of comparison between these different methods could be that of their behavior in regards to the value of c. This behavior could be monitored through two measures; (1) accuracy and (2) convergence.

Accuracy represents the closeness of a solution to the optimal solution and has been formulated through W in the following manner:

$$A = exp(-\alpha|W|) \tag{6.27}$$

where α is a constant chosen so as to make the final values comparable. By this method, overall accuracy is only equal to 1 when $W = 0$ and is otherwise less than 1.

Convergence on the other hand represents the frequency of a method converging to a settling point regardless of how optimal this final stop may be. There is thus no real formula for this measure and it could only be calculated after running each specific algorithm many times.

It would be interesting to see how each of these methods change their accuracy and convergence with a change in the value of c as it seems to play a critical role in the formation of the objective function.

6.5.4 Simulation Settings

We run a simulation over the channel $(U_1, U_2) \to X \to Y$ where I have assumed that $|U_1| = |U_2| = 256$ and $|X| = 512$ and $|Y| = 20$. The elements within the probabilistic channel $p(x|u_1, u_2)$ are randomly chosen and then normalized to account for the probabilistic nature of the matrix. Furthermore $p(u_1)$ and $p(u_2)$ are randomly chosen and normalized to 1.

While discussing the non-optimality measure, the cost function is chosen to be $c = 10$ and to account for as many scenarios as possible, a mixed range of $\beta \in \{1, 2, ..., 20\}$ and $\gamma \in \{1, 2, ..., 20\}$ is chosen and optimization is run over all 400 possible cases.

While discussing the accuracy and convergence behaviors, it is assumed that $\beta = \gamma = 10, \alpha = 5$ and that $c \in \{1, ..., 3000\}$.

6.5.5 Simulation Results

It turns out that the final overall objective function attained by either the R-ADMM or OR-ADMM algorithm are much less than the objective function developed by IB as witnessed in Fig. 6.1.

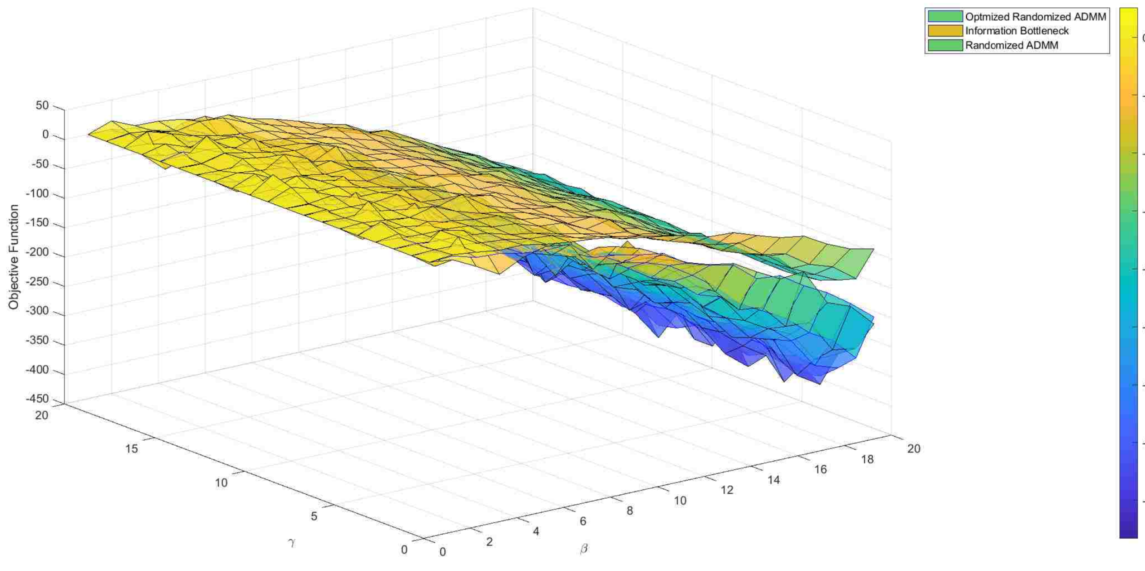

FIGURE 6.1. Objective Function for R-ADMM, OR-ADMM and IB vs β and γ

Furthermore, Fig. 6.2 offers a 3D representation of the measure of non-optimality versus different values of β and γ. As can be witnessed, the amount of non-optimality offered by the Information Bottleneck method is much higher than those offered by the ADMM based methods thus further proving the superiority of the ADMM-based method.

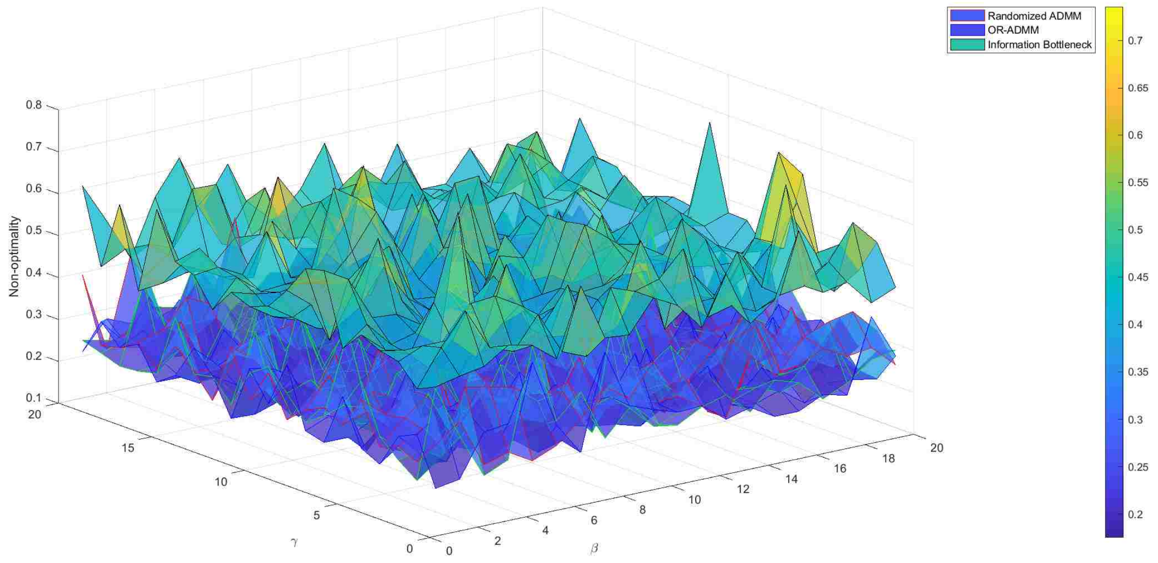

FIGURE 6.2. Non-optimality vs β and γ for R-ADMM, OR-ADMM and IB

Finally, I conclude this section by offering Fig. 6.3 which represent the behavior of the 3 algorithms as far as their accuracy and convergence rate are concerned as the value of c grows larger.

As can be seen, the accuracy of the 2 ADMM-based methods are quite the same for different values of c and better than that offered by IB method. What is more interesting is how as c grows larger, the accuracy seems to deteriorate in all methods. This is because as c grows larger, the original objective function goes through a larger change to form na new $\mathcal{L}$, $\mathcal{L}^{'}$ and $\mathcal{L}^{''}$ respectively. Then however, minimizing these 3 objective functions is no longer as closely comparable with minimizing the original objective function $\mathcal{L}$.

Furthermore, it can be seen that as c grows larger, the convergence rate goes up in all ADMM-based algorithms. The reason for such behavior can once again be found in the definition of new objective functions $\mathcal{L}, \mathcal{L}'$ and $\mathcal{L}''$. As c grows larger, the quadratic section of these objective functions becomes more eminent and as a result the overall objective function tends to behave more like a quadratic function thus resulting in a higher convergence rate.

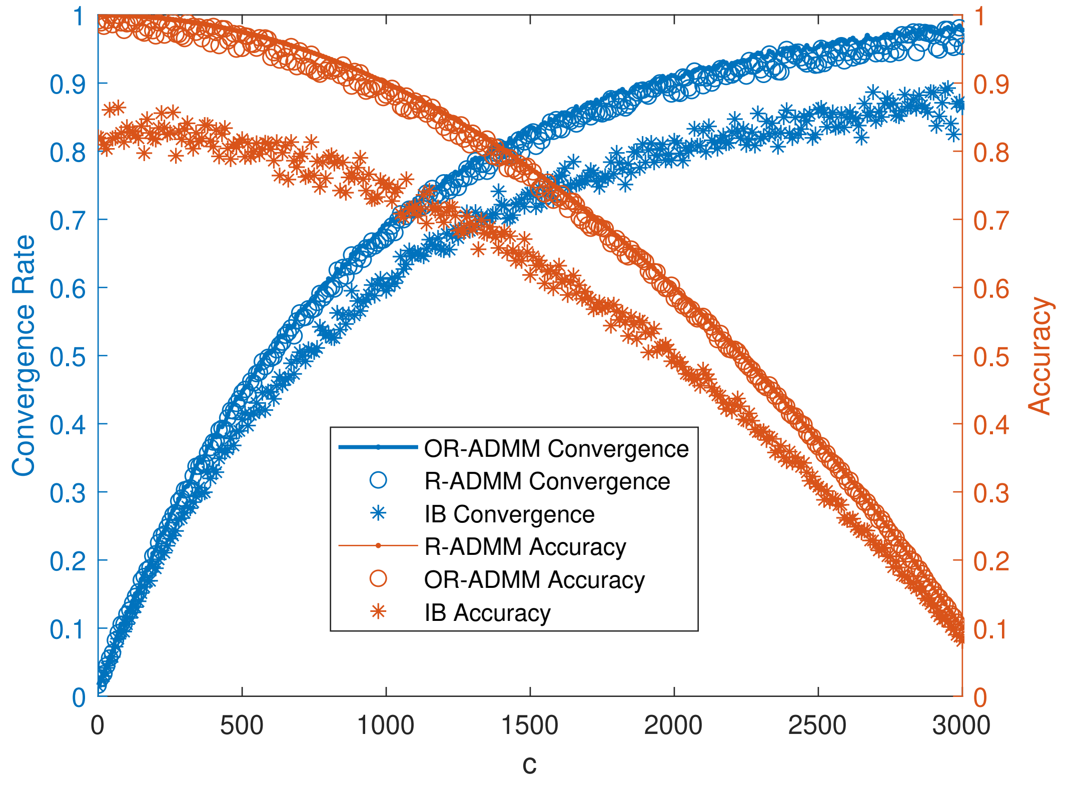

FIGURE 6.3. Convergence Rate and Accuracy vs c variable for Algorithms

So far, I have managed to indicate the superiority of ADMM-based methods over IB. However there is one more measure which I have so far neglected; speed of convergence. It turns out that while running these two algorithms will result in almost the same accuracy and convergence rate, the speed of convergence in Algorithm 3 is much higher than that of Algorithm 2. This observation is justifiable seeing as how in Algorithm 2, every run of gradient descent attempts to solve a non-convex problem (overall 4 possible gradient descents in every iterations), however in Algorithm 3, I have managed to decrease the number of required gradient descents over non-convex functions to 3, thus reducing the delay in every iteration. Over a large number of repetitions, this minor change in the number of gradient descent runs results in an undeniable superiority for Algorithm 3 in comparison with Algorithm 2.

Thus, overall I could conclude that Algorithm 3 offers the best objective function with the best rate and speed of convergence in comparison with all other ADMM-based algorithms and the IB method.

6.6 Conclusions

In this chapter, I revised a well-known machine learning problem where the goal is to reveal as much information about a segment of the data while keeping another segment of it, relatively hidden. To do so, I modeled the settings in the format of a Lagrange multiplier problem. Initially motivated by [9]'s work, I opted to carry out the IB method over the problem. However, since I had already shown in [14] that such a method might not be optimal, I chose to develop ADMM-based algorithms for solving the same problem. 2 different methods were developed and their respective resulting objective functions were calculated. Then through many examinations and numerical results, I identified the optimal solution of the 2 and indicated its many advantages over the previous methods discussed in works such as [15].

Chapter 7
Proposed Works: Empirical Data Analysis

Thus far, in both Chapters 5 and 6 I have been concerned about leaked information assuming a full knowledge of the true distribution between the input and the median (U and X variables) in the former and between deliverable and private features and the median ((U_1, U_2) and X variables) in the latter. However, such a knowledge of the true distribution might not be feasible in real life applications where I only have access to a limited set of data and thus can only access an empirical distribution. I then would like to answer the question "How closely do the results gained from this empirical data distribution resemble those of an ideally true data distribution?" Rather, if I only have access to an empirical distribution of the dataset, will treating it as the true distribution of the dataset be misleading or could it be justified?

7.1 Recent Works in the Field

A version of this question was recently acknowledged in [16] wherein a problem quite similar to the original Information Bottleneck was challenged. In a channel $U \to X \to Y$, it is assumed that only empirical knowledge of the probability distributions $\tilde{P}_U(u)$ and $\tilde{P}_{X|U}(x|u)$ are known, a priori. . In this channel, U and X represent the private and non-private features of the data which are connected to one another through the conditional channel $P_{X|U}(x|u)$.

Then a randomized mapping aka "privacy mechanism" $P_{Y|X}(y|x)$ is applied to the non-private features to produce a perturbed version of the original data which satisfies a series of privacy and utility guarantees over the empirical distributions. These guarantees usually have to do with some form of information about the private and non-private features through the output variable Y and protecting or revealing these types of information to an adversary.

7.1.1 Real-world Example

As an example [63], assume that a dataset containing the information about 2500 prisoners and whether or not they were offered parole is available. This data sheet may contain some sensitive features such as the name of the prisoner (U is the name). There are some other features in this data sheet which may not be necessarily sensitive features but could be related to such sensitive features for example race, gender or zip code through an empirical conditional channel $P_{X|U}(x|u)(X$ represents these features). Thus, simply removing the private feature (names) may not hide the sensitive information since they could still be accurately inferred through other features. Thus, a randomized mapping $P_{Y|X}(y|x)$ is applied to non-private features to help make such private features less accessible. The output of this mapping is then protected through a semi-design of an IB problem.

Then, the main question is "How strong do the results of this design for a limited (and thus empirical) number of samples hold true for a completely new set of data?"

So far, the problem formulation in [16] is quite similar to that of an Information Bottleneck problem. However, rather than simply designing an optimal privacy mechanism given the empirical distribution at hand and using mutual information as a measure of revealed information as was done in Chapters 5 and 6, [16] sought out to see the required conditions on functions of revealed information for an empirical distribution to be closely applicable for a true distribution as well.

7.1.2 Results

[16] showed that for a certain family of functions representing leaked information, it could be shown that the results for the design of a privacy mechanism based upon limited empirical data work just as well for those based on true data within

an acceptable bound. These families of functions include α-leakage [64], Arimoto's mutual information of order $\alpha \in (0, \infty)$ [65, 66], the maximal α-leakage [64] and Sibson's mutual information of order $\alpha \in (0, \infty)$ [69].

This was accomplished by (1) rewriting the overall problem formulation using each of the above functions as measures of leaked information (2) utilizing the definition of Lipschitz continuity for each of these functions to show that they are upper bounded and (3) utilizing the definition of Large Distribution Bounds for distribution estimation.

7.1.3 Future Works

Although the above results may sound like fine findings in my problem scenario as well, unfortunately [16] specifically showed that mutual information does not satisfy the Lipschitz continuity condition and thus cannot be used as a measure of revealed information for real-world problems where I am dealing with empirical distributions.

As a result, in order to implement my ADMM-based methods into real-world applications, I will need to start employing one of the above functions of α-leakage including Arimoto's mutual information of order $\alpha \in (0, \infty)$, the maximal α-leakage and Sibson's mutual information of order $\alpha \in (0, \infty)$ as a measure of revealed information and rethink my algorithm from there.

In addition, for the problem under the model of Markov Chain $(U_1, U_2) \rightarrow X \rightarrow Y$, I want to retain the capacity of inference using Y about U_1, while keeping the leakage on U_2 minimum, i.e. $f(U_1|X) - f(U_1|Y)$ should be minimized, while $f(U_2|Y)$ satisfies a lower bound , where the function f could be a metric adopted as in [16]. With training data only, I could face a problem of defining both (U_1, U_2), and the conditional $p(X|U_1, U_2)$, while keeping the marginal of X unchanged, by seeking a properly defined set of latent variables (U_1, U_2) under precedent constraints.

Appendix A: Supplementary Material for Chapter 3

A.1. Theorem III.1

Proof. For an easier understanding, my proof of the theorem is broken into two sections:

Sufficient Conditions for $\mathbf{F_1(S_j)}$:

Starting for $F_1(S_j)$, by rewriting Eq. (3.8) I will have:

$$\begin{gathered}(\alpha_A + \alpha_x) f(A + \{x\}) + \lambda(\alpha_A + \alpha_x) \log(\alpha_A + \alpha_x) \\ -\alpha_A f(A) - \lambda \alpha_A \log(\alpha_A) \geq \\ (\alpha_A + \alpha_{BA} + \alpha_x) f(B + \{x\}) \\ +\lambda(\alpha_A + \alpha_{BA} + \alpha_x) \log(\alpha_A + \alpha_{BA} + \alpha_x) \\ -(\alpha_A + \alpha_{BA}) f(B) - \lambda(\alpha_A + \alpha_{BA}) \log(\alpha_A + \alpha_{BA}) \end{gathered} \tag{7.1}$$

I can then factorize the inequality as

$$\begin{gathered}\alpha_{BA}[-f(B + \{x\}) + f(B) - \lambda \log(\alpha_A + \alpha_{BA} + \alpha_x) \\ +\lambda \log(\alpha_A + \alpha_{BA})] + \alpha_x[f(A + \{x\}) \\ -f(B + \{x\}) + \lambda \log(\alpha_A + \alpha_x) - \lambda \log(\alpha_A + \alpha_{BA} + \alpha_x)] \\ +\alpha_A[f(A + \{x\}) - f(B + \{x\}) - f(A) + f(B) \\ +\lambda \log(\alpha_A + \alpha_x) - \lambda \log(\alpha_A + \alpha_{BA} + \alpha_x) \\ -\lambda \log(\alpha_A) + \lambda \log(\alpha_A + \alpha_{BA})] \geq 0 \end{gathered} \tag{7.2}$$

Using g and q set functions, I can then rewrite the factorization as:

$$\alpha_{BA}[-g(B+\{x\},B)+\lambda\log[\frac{1}{1+\frac{\alpha_x}{\alpha_A+\alpha_{BA}}}]]$$
$$+\alpha_x[-g(B+\{x\},A+\{x\})+\lambda\log[\frac{1}{1+\frac{\alpha_{BA}}{\alpha_A+\alpha_x}}]]$$
$$+\alpha_A[-q(B+\{x\},B,A+\{x\},A)+\lambda\log\frac{1+\frac{\alpha_{BA}}{\alpha_A}}{1+\frac{\alpha_{BA}}{\alpha_A+\alpha_x}}\geq 0 \tag{7.3}$$

Since the above inequality needs to hold true for all possible sets of $A\subseteq B$, $x\notin B$, I aim to determine the maximal amount enforced by the above set of inequalities. The first factorization results in two inequalities: (1) $g(C,D)\leq 0$ and (2) $|g(C,D)|\geq\lambda max(\log(1+\frac{\alpha_x}{\alpha_B}))$ for all sets $D\subseteq C$. To find the maximum of such a limit, I need to impose the one item with highest probability to $\{x\}$ and assume the one lowest probability item to set B. Then the above inequality is maximized.

The second factorization results in two inequalities: (1) $g(C,D)\leq 0$ and (2) $|g(C,D)|\geq\lambda max(\log(1+\frac{\alpha_{BA}}{\alpha_A+\alpha_x}))$ for all sets $D\subseteq C$. To find the maximum of such a limit, I need to impose the item with lowest probability to $\{x\}$ and that $\alpha_A=0$ and then have $\alpha_{BA}=1-\alpha_x=\alpha_B$.

Once again, it is important to note that by doing so, I am removing the possibility of $\alpha_A=0$ and $\alpha_x=0$ as a case. In other words, I am assuming that $\alpha_A+\alpha_x=0$ is not a scenario I need to discuss. I will now demonstrate why such an assumption is accurate.

When $\alpha_A+\alpha_x=0$, I can rewrite inequality (3.8) in the following format:

$$0\geq\alpha_{BA}f(B)-\alpha_{BA}f(B)=0 \tag{7.4}$$

which is always true.

The third factorization could be simplified. The logarithm argument consists of a nominator greater than denominator thus resulting in the overall logarithm argument to be positive. I thus only need to impose that $q(C, C_1, D, D_1) \leq 0$ for all sets $C_1 \subseteq C, D_1 \subseteq D$ and $D \subseteq C$.

It then follows that the probability of each item is sorted in a decreasing manner as $\pi_1, ..., \pi_M$, I can write the set of 3 sufficient conditions for submodularity of set function $F_1(S_j), j = 1, ..., N$ as (1) $g(C, D) \leq 0$ (2) $q(C, C_1, D, D_1) \leq 0$ (3) $|g(C, D)| \geq \lambda \log\left[max(1 + \frac{\pi_1}{\pi_M}, 1 + \frac{1-\pi_M}{\pi_M})\right] = \lambda \log\left(\frac{1}{\pi_M}\right)$ for all sets $C_1 \subseteq C, D_1 \subseteq D$ and $D \subseteq C$.

Note 1: It is important to note that any maximization is carried out by assuming that the lower bound cannot go to ∞. However there are times when specific set allocations could result in denominators being equal to 0. Such set allocations then need to be addressed separately by rewriting Inequality (3.8) and finding their specific sufficient conditions.

In this proof, I have two denominators α_B and $\alpha_A + \alpha_x$. I need to address each specific condition under which either of these are equal to 0 to find their specific sufficient conditions.

First, I assume that $\alpha_B = 0$. This in turn means that $B = \emptyset$. Furthermore, since $A \subseteq B$, I can conclude that $\alpha_A = \alpha_B = 0$ and then rewrite inequality (3.8) in the following format:

$$\alpha_x f(\{x\}) \geq \alpha_x f(\{x\}) \tag{7.5}$$

which is always true.

Second, I assume that $\alpha_A + \alpha_x = 0$; however, this could never occur seeing as how $\alpha_x > 0$. Thus, the second possibility does not need to be further checked.

Sufficient Conditions for $\mathbf{F_2(S_j)}$: I now follow the same method for $F_2(S_j)$, $j = 1, ..., N$ by rewriting Eq. (3.8):

$$\begin{gathered}(1-\alpha_A-\alpha_x)f(S-A-\{x\})+\\ \lambda(1-\alpha_A-\alpha_x)\log(1-\alpha_A-\alpha_x)-(1-\alpha_A)f(S-A)\\ -\lambda(1-\alpha_A)\log(1-\alpha_A)\geq\\ (1-\alpha_A-\alpha_{BA}-\alpha_x)f(S-B-\{x\})\\ +\lambda(1-\alpha_A-\alpha_{BA}-\alpha_x)\log(1-\alpha_A-\alpha_{BA}-\alpha_x)\\ -(1-\alpha_A-\alpha_{BA})f(S-B)-\\ \lambda(1-\alpha_A-\alpha_{BA})\log(1-\alpha_A-\alpha_{BA}) \end{gathered} \tag{7.6}$$

I can then factorize the inequality as

$$\begin{gathered}\alpha_{BA}[f(S-B-\{x\})-f(S-B)\\ -\lambda\log(1+\frac{\alpha_x}{1-\alpha_A-\alpha_{BA}-\alpha_x})]\\ +\alpha_x[-f(S-A-\{x\})+f(S-B-\{x\})\\ -\lambda\log(1+\frac{\alpha_{BA}}{1-\alpha_A-\alpha_x-\alpha_{BA}})]\\ +\alpha_A[-f(S-A-\{x\})+f(S-B-\{x\})+f(S-A)\\ -f(S-B)+\lambda\log(\frac{1-\alpha_A}{1-\alpha_A-\alpha_x}\frac{1-\alpha_A-\alpha_{BA}-\alpha_x}{1-\alpha_A-\alpha_{BA}})]\\ +1[f(S-A-\{x\})-f(S-B-\{x\})-f(S-A)\\ +f(S-B)+\lambda\log(\frac{1-\alpha_A-\alpha_x}{1-\alpha_A}\frac{1-\alpha_A-\alpha_{BA}}{1-\alpha_A-\alpha_{BA}-\alpha_x})]\\ \geq 0\end{gathered} \tag{7.7}$$

Using g and q set functions, I can rewrite the factorization as:

$$\begin{aligned}
&\alpha_{BA}[-g(S-B,S-B-\{x\}) \\
&-\lambda\log(1+\frac{\alpha_x}{1-\alpha_A-\alpha_{BA}-\alpha_x})] \\
&+\alpha_x[-g(S-A-\{x\},S-B-\{x\}) \\
&-\lambda\log(1+\frac{\alpha_{BA}}{1-\alpha_A-\alpha_x-\alpha_{BA}})] \\
&+\alpha_A[q(S-A,S-A-\{x\},S-B,S-B-\{x\}) \\
&+\lambda\log(\frac{1-\alpha_A}{1-\alpha_A-\alpha_x}\frac{1-\alpha_A-\alpha_{BA}-\alpha_x}{1-\alpha_A-\alpha_{BA}})] \\
&+1[-q(S-A,S-A-\{x\},S-B,S-B-\{x\}) \\
&+\lambda\log(\frac{1-\alpha_A-\alpha_x}{1-\alpha_A}\frac{1-\alpha_A-\alpha_{BA}}{1-\alpha_A-\alpha_{BA}-\alpha_x})]\geq 0
\end{aligned}\tag{7.8}$$

Since the above inequality needs to hold true for all possible sets of $A \subseteq B$, $x \notin B$, I aim to determine the maximal amount enforced by the above set of inequalities. The first factorization results in two inequalities: (1) $g(C,D) \leq 0$ and (2) $|g(C,D)| \geq \lambda\log(1+\frac{\alpha_x}{1-\alpha_B-\alpha_x})$ for all sets $D \subseteq C$. To find the maximum of such a limit, I need to impose the one item with highest probability to $\{x\}$ and assume that α_B is maximal while still less than $1-\alpha_x$. This limit is imposed so that the denominator is not equal to 0.

The second factorization results in two inequalities: (1) $g(C,D) \leq 0$ and (2) $|g(C,D)| \geq \lambda max(\log(1+\frac{\alpha_{BA}}{1-\alpha_B-\alpha_x}))$ for all sets $D \subseteq C$. To find the maximum of such a limit, I once again need to impose the one item with highest probability to $\{x\}$ and assume that $\alpha_A = 0$ and α_B is maximal while still less than $1-\alpha_x$. This limit is once again imposed so that the denominator is not equal to 0.

The third and forth factorizations could be simplified. I could write them as

$$\begin{gathered}q(S-A, S-A-\{x\}, S-B, S-B-\{x\})[-1+\alpha_A]+\\ \lambda\alpha_A \log\left(\frac{1-\alpha_A}{1-\alpha_A-\alpha_x}\frac{1-\alpha_A-\alpha_{BA}-\alpha_x}{1-\alpha_A-\alpha_{BA}}\right)\\ +\lambda \log\left(\frac{1-\alpha_A}{1-\alpha_A-\alpha_{BA}}\frac{1-\alpha_A-\alpha_x}{1-\alpha_A-\alpha_{BA}-\alpha_x}\right) \geq 0\end{gathered} \tag{7.9}$$

Depending on whether the sum of logarithmic arguments on the LHS is positive or negative, I can write two inequalities:

$$\begin{gathered}(1) \quad \lambda\alpha_A \log\left(\frac{1-\alpha_A}{1-\alpha_A-\alpha_x}\frac{1-\alpha_A-\alpha_{BA}-\alpha_x}{1-\alpha_A-\alpha_{BA}}\right)\\ +\lambda \log\left(\frac{1-\alpha_A}{1-\alpha_A-\alpha_{BA}}\frac{1-\alpha_A-\alpha_x}{1-\alpha_A-\alpha_{BA}-\alpha_x}\right) \geq 0 \rightarrow\\ q(S-A, S-A-\{x\}, S-B, S-B-\{x\})[-1+\alpha_A]+\\ \lambda\alpha_A \log\left(\frac{1-\alpha_A}{1-\alpha_A-\alpha_x}\frac{1-\alpha_A-\alpha_{BA}-\alpha_x}{1-\alpha_A-\alpha_{BA}}\right)\\ +\lambda \log\left(\frac{1-\alpha_A}{1-\alpha_A-\alpha_{BA}}\frac{1-\alpha_A-\alpha_x}{1-\alpha_A-\alpha_{BA}-\alpha_x}\right) \geq\\ q(S-A, S-A-\{x\}, S-B, S-B-\{x\})[-1+\alpha_A]\\ +\lambda\alpha_A \log\left(\frac{1-\alpha_A}{1-\alpha_A-\alpha_{BA}}\right) \overset{?}{\geq} 0\end{gathered} \tag{7.10}$$

where I have simply summed the two logarithmic arguments. It then turns out that the logarithmic argument is always positive seeing as how the nominator of the fraction inside it is greater than the denominator. It thus follows that I merely need to impose $q(C, C_1, D, D_1) \leq 0$ for all sets $C_1 \subseteq C, D_1 \subseteq D$ and $D \subseteq C$ to help above inequality hold true. A second scenario dictates when:

$$\begin{aligned}
&(1) \quad \lambda\alpha_A \log\left(\frac{1-\alpha_A}{1-\alpha_A-\alpha_x}\frac{1-\alpha_A-\alpha_{BA}-\alpha_x}{1-\alpha_A-\alpha_{BA}}\right)\\
&+\lambda \log\left(\frac{1-\alpha_A}{1-\alpha_A-\alpha_{BA}}\frac{1-\alpha_A-\alpha_x}{1-\alpha_A-\alpha_{BA}-\alpha_x}\right) \leq 0 \rightarrow\\
&q(S-A, S-A-\{x\}, S-B, S-B-\{x\})[-1+\alpha_A]+\\
&\lambda\alpha_A \log\left(\frac{1-\alpha_A}{1-\alpha_A-\alpha_x}\frac{1-\alpha_A-\alpha_{BA}-\alpha_x}{1-\alpha_A-\alpha_{BA}}\right)\\
&+\lambda \log\left(\frac{1-\alpha_A}{1-\alpha_A-\alpha_{BA}}\frac{1-\alpha_A-\alpha_x}{1-\alpha_A-\alpha_{BA}-\alpha_x}\right) \geq\\
&q(S-A, S-A-\{x\}, S-B, S-B-\{x\})[-1+\alpha_A]\\
&+\lambda \log\left(\frac{1-\alpha_A}{1-\alpha_A-\alpha_{BA}}\right) \overset{?}{\geq} 0 \qquad (7.11)
\end{aligned}$$

where I have once again simply summed the two logarithmic arguments. It again turns out that the logarithmic argument is always positive seeing as how the nominator of the fraction inside it is greater than the denominator. It once more thus follows that I merely need to impose $q(C, C_1, D, D_1) \leq 0$ for all sets $C_1 \subseteq C, D_1 \subseteq D$ and $D \subseteq C$ to help above inequality hold true.

Note 2: Once again, note that any maximization is carried out by assuming that the lower bound cannot go to ∞. However there are times when specific set allocations could result in denominators being equal to 0. Such set allocations then need to be addressed separately by rewriting Inequality (3.8) and finding their specific sufficient conditions.

In this proof, I have one denominators $1-\alpha_B-\alpha_x$ appearing twice. I need to address the specific condition under which this amount is equal to 0. This could only occur when $B = \{x\}^C$. In such cases, I can rewrite inequality (3.8) in the following format:

$$
\begin{gathered}
(1-\alpha_A-\alpha_x)f(S-A-\{x\})+ \\
\lambda(1-\alpha_A-\alpha_x)\log(1-\alpha_A-\alpha_x)-(1-\alpha_A)f(S-A) \\
-\lambda(1-\alpha_A)\log(1-\alpha_A)\geq-(1-\alpha_A-\alpha_{BA})f(S-B) \\
-\lambda(1-\alpha_A-\alpha_{BA})\log(1-\alpha_A-\alpha_{BA})\rightarrow \\
\alpha_{BA}\{-f(S-B)-\lambda\log(1-\alpha_A-\alpha_{BA})\}+ \\
\alpha_x\{-f(S-A-\{x\})-\lambda\log(1-\alpha_A-\alpha_{BA})\}+ \\
\alpha_A\{-f(S-A-\{x\})+f(S-A)-f(S-B)- \\
\lambda\log(1-\alpha_A-\alpha_x)+\lambda\log(1-\alpha_A) \\
-\lambda\log(1-\alpha_A-\alpha_{BA})\}+1\{f(S-A-\{x\})-f(S-A)+ \\
f(S-B)+\lambda\log(1-\alpha_A-\alpha_x)-\lambda\log(1-\alpha_A) \\
+\lambda\log(1-\alpha_A-\alpha_{BA})\}\geq 0
\end{gathered}
\tag{7.12}
$$

By adding and removing three factors of $f(S-B-\{x\})$, $\lambda\log(1-\alpha_A-\alpha_x)$ and $\lambda\log(1-\alpha_A-\alpha_{BA})$ to each of the factorizations above and using the definitions of g and q functions as previously, I can simplify the above inequality in the following format:

$$\alpha_{BA}\{-g(S-B,S-B-\{x\})+\lambda\log(1-\alpha_A-\alpha_x)\}+$$
$$\alpha_x\{-g(S-A-\{x\},S-B-\{x\})+\lambda\log(1-\alpha_A-\alpha_{BA})\}$$
$$+\alpha_A\{q(S-A,S-B,S-A-\{x\},S-B-\{x\})+$$
$$\lambda\log(1-\alpha_A)\}+$$
$$1\{-q(S-A,S-B,S-A-\{x\},S-B-\{x\})\}$$
$$-\lambda\log(1-\alpha_A)\}+$$
$$\{\lambda\log(1-\alpha_A-\alpha_x)+\lambda\log(1-\alpha_A-\alpha_{BA})\}$$
$$\{-\alpha_{BA}-\alpha_x-\alpha_A+1\}\geq 0 \tag{7.13}$$

Using the fact that $\alpha_{BA}+\alpha_x+\alpha_A=1$ and merging the 3^{rd} and 4^{th} factorizations together, I will have:

$$\alpha_{BA}\{-g(S-B,S-B-\{x\})+\lambda\log(1-\alpha_A-\alpha_x)\}+$$
$$\alpha_x\{-g(S-A-\{x\},S-B-\{x\})+$$
$$\lambda\log(1-\alpha_A-\alpha_{BA})\}+(1-\alpha_A)$$
$$\{-q(S-A,S-B,S-A-\{x\},S-B-\{x\})$$
$$-\lambda\log(1-\alpha_A)\}\geq 0 \tag{7.14}$$

which would hold true as long as (1) $g(C,D)\leq 0$ for all sets C and (2) $|g(C,D)|\geq\lambda max(\log\frac{1}{\alpha_x})$ and (3) $q(C,C_1,D,D_1)\leq 0$ for all sets $C_1\subseteq C, D_1\subseteq D$ and $D\subseteq C$. It could be perceived that this specific set allocation demands a different lower bound for the absolute value of the first order difference of function f over sets.

Thus 3 sufficient conditions for submodularity of $F_2(S_j), j=1,...,N$ are developed: (1) $g(C,D)\leq 0$ (2) $q(C,C_1,D,D_1)\leq 0$ (3) $|g(C,D)|\geq\lambda\log(\frac{1}{\pi_M})$ for all sets $C_1\subseteq C, D_1\subseteq D$ and $D\subseteq C$.

Finally, the intersection of the two sets of sufficient conditions for submodularity of $F_1(S_j)$ and $F_2(S_j)$ gives us the sufficient conditions for submodularity of $F(S_j)$ which turns out to be the same as either of theirs.

□

A.2. Lemma III.2

Proof. For the specific case $N = 2$:

$$\begin{aligned} F(A) = \alpha_A f(A) + (1-\alpha_A) f(S-A) + \alpha_A \log(\alpha_A) \\ +(1-\alpha_A)\log(1-\alpha_A) \end{aligned} \tag{7.15}$$

In such a case, diminishing return property requires that

$$\begin{gathered} (\alpha_A+\alpha_x) f(A+\{x\}) + (1-\alpha_A-\alpha_x) f(S-A-\{x\}) \\ +(\alpha_A+\alpha_x)\log(\alpha_A+\alpha_x) \\ +(1-\alpha_A-\alpha_x)\log(1-\alpha_A-\alpha_x) - (\alpha_A) f(A) \\ -(1-\alpha_A) f(S-A) - (\alpha_A)\log(\alpha_A) \\ -(1-\alpha_A)\log(1-\alpha_A) \geq \\ (\alpha_A+\alpha_{BA}+\alpha_x) f(B+\{x\}) \\ +(1-\alpha_A-\alpha_{BA}-\alpha_x) f(S-B-\{x\}) \\ +(\alpha_A+\alpha_{BA}+\alpha_x)\log(\alpha_A+\alpha_{BA}+\alpha_x) \\ +(1-\alpha_A-\alpha_{BA}-\alpha_x)\log(1-\alpha_{BA}-\alpha_A-\alpha_x) \\ -(\alpha_B) f(B) - (1-\alpha_A-\alpha_{BA}) f(S-B) \\ -(\alpha_A+\alpha_{BA})\log(\alpha_A+\alpha_{BA}) \\ -(1-\alpha_A-\alpha_{BA})\log(1-\alpha_A-\alpha_{BA}) \end{gathered} \tag{7.16}$$

I then factorize this inequality in the following manner:

$$\begin{gathered}
\alpha_{BA}\{-f(B+\{x\})+f(B)+f(S-B-\{x\})-f(S-B) \\
+\lambda\log(\frac{\alpha_A+\alpha_{BA}}{\alpha_A+\alpha_{BA}+\alpha_x}\frac{1-\alpha_A-\alpha_{BA}-\alpha_x}{1-\alpha_A-\alpha_{BA}})\}+ \\
\alpha_x\{f(A+\{x\})-f(B+\{x\})-f(S-A-\{x\}) \\
+f(S-B-\{x\})+ \\
\lambda\log(\frac{\alpha_A+\alpha_x}{\alpha_A+\alpha_x+\alpha_{BA}}\frac{1-\alpha_A-\alpha_x-\alpha_{BA}}{1-\alpha_A-\alpha_x})\} \\
+\alpha_A\{f(A+\{x\})-f(B+\{x\})-f(S-A-\{x\}) \\
+f(S-B-\{x\})-f(A)+f(B)+f(S-A)-f(S-B) \\
+\log(\frac{\alpha_A+\alpha_x}{\alpha_A+\alpha_x+\alpha_{BA}}\frac{1-\alpha_A-\alpha_x-\alpha_{BA}}{1-\alpha_A-\alpha_x}\frac{\alpha_A+\alpha_{BA}}{\alpha_A}\frac{1-\alpha_A}{1-\alpha_A-\alpha_{BA}})\} \\
+\{f(S-A-\{x\})-f(S-B-\{x\})-f(S-A) \\
+f(S-B)+\lambda\log(\frac{1-\alpha_A-\alpha_x}{1-\alpha_A-\alpha_x-\alpha_{BA}}\frac{1-\alpha_A-\alpha_{BA}}{1-\alpha_A})\} \\
\geq 0 \qquad (7.17)
\end{gathered}$$

I then use the definitions of set functions g and q to simplify the above inequality:

$$\begin{aligned}
&\alpha_{BA}\{-g(B+\{x\},B)-g(S-B,S-B-\{x\}) \\
&+\lambda\log(\frac{\alpha_A+\alpha_{BA}}{\alpha_A+\alpha_{BA}+\alpha_x}\frac{1-\alpha_A-\alpha_{BA}-\alpha_x}{1-\alpha_A-\alpha_{BA}})\}+ \\
&\alpha_x\{-g(B+\{x\},A+\{x\})-g(S-A-\{x\},S-B-\{x\}) \\
&+\lambda\log(\frac{\alpha_A+\alpha_x}{\alpha_A+\alpha_x+\alpha_{BA}}\frac{1-\alpha_A-\alpha_x-\alpha_{BA}}{1-\alpha_A-\alpha_x})\} \\
&+\alpha_A\{-q(B+\{x\},B,A+\{x\},A) \\
&+q(S-A,S-A-\{x\},S-B,S-B-\{x\})+\lambda \\
&\log(\frac{\alpha_A+\alpha_x}{\alpha_A+\alpha_x+\alpha_{BA}}\frac{1-\alpha_A-\alpha_x-\alpha_{BA}}{1-\alpha_A-\alpha_x}\frac{\alpha_A+\alpha_{BA}}{\alpha_A}\frac{1-\alpha_A}{1-\alpha_A-\alpha_{BA}})\} \\
&+\{-q(S-A,S-A-\{x\},S-B,S-B-\{x\}) \\
&+\lambda\log(\frac{1-\alpha_A-\alpha_x}{1-\alpha_A-\alpha_x-\alpha_{BA}}\frac{1-\alpha_A-\alpha_{BA}}{1-\alpha_A})\} \\
&\geq 0
\end{aligned} \tag{7.18}$$

I aim to find sufficient conditions so that the above inequality can hold true. The first factorization is the coefficients of $\alpha_{BA} \geq 0$. I thus need to ascertain that the coefficient is also positive. To do so, I impose that

$$\begin{aligned}
&-2g(C,D)+\lambda\log(\frac{\alpha_A+\alpha_{BA}}{\alpha_A+\alpha_{BA}+\alpha_x}\frac{1-\alpha_A-\alpha_{BA}-\alpha_x}{1-\alpha_A-\alpha_{BA}})\} \\
&\geq 0
\end{aligned} \tag{7.19}$$

for all possible sets $D \subseteq C$. I thus need to impose two sufficient conditions:

$$\begin{aligned}
&(1) \quad g(C,D) \leq 0, \forall C \\
&(2) \quad 2|g(C,D)| \geq \\
&max(|\lambda\log(\frac{\alpha_A+\alpha_{BA}}{\alpha_A+\alpha_{BA}+\alpha_x}\frac{1-\alpha_A-\alpha_{BA}-\alpha_x}{1-\alpha_A-\alpha_{BA}})\}|)
\end{aligned} \tag{7.20}$$

The second factorization is the coefficients of $\alpha_x \geq 0$. I once again need to ascertain that the coefficient is also positive. To do so, I impose that

$$\begin{gathered}(1) \quad g(C,D) \leq 0, \forall C \\ (2) \quad 2|g(C,D)| \geq \\ max(|\lambda \log(\frac{\alpha_A + \alpha_x}{\alpha_A + \alpha_x + \alpha_{BA}} \frac{1 - \alpha_A - \alpha_x - \alpha_{BA}}{1 - \alpha_A - \alpha_x})|) \end{gathered} \tag{7.21}$$

for all sets $D \subseteq C$. I could see that the RHS of the secondary condition could be written as:

$$\lambda max(\log((1 + \frac{\alpha_{BA}}{\alpha_A + \alpha_x})(1 + \frac{\alpha_{BA}}{1 - \alpha_A - \alpha_x}))) = K \tag{7.22}$$

It is clear to see that the two fractions could not be maximal at the same time seeing as how (1) the maximal occurs when denominator of each is close to zero and (2) their denominators have opposing behavior. It thus turns out that the RHS as indicated in Eq. (7.22) is limited by:

$$K < \lambda \log(max(1 + \frac{\alpha_{BA}}{\alpha_A + \alpha_x})max(1 + \frac{\alpha_{BA}}{1 - \alpha_A - \alpha_x})) \tag{7.23}$$

The first fraction is maximized when $\alpha_A = 0$, $\alpha_x = \pi_M$ and $\alpha_{BA} = 1 - \pi_M$ while the second fraction can never be as high. It thus follows that:

$$K < \lambda \log(\frac{1}{\pi_M})^2 \tag{7.24}$$

The 3^{rd} and 4^{th} factorizations could be merged together in the following manner:

$$\begin{gathered}[-1 + \alpha_A]q(S - A, S - A - \{x\}, S - B, S - B - \{x\}) - \alpha_A q(B + \{x\}, B, A + \{x\}, A) \\ +\lambda\{\alpha_A \log(\frac{\alpha_A + \alpha_x}{\alpha_A + \alpha_x + \alpha_{BA}} \frac{1 - \alpha_A - \alpha_x - \alpha_{BA}}{1 - \alpha_A - \alpha_x} \frac{\alpha_A + \alpha_{BA}}{\alpha_A} \frac{1 - \alpha_A}{1 - \alpha_A - \alpha_{BA}}) \\ + \log(\frac{1 - \alpha_A - \alpha_x}{1 - \alpha_A - \alpha_x - \alpha_{BA}} \frac{1 - \alpha_A - \alpha_{BA}}{1 - \alpha_A})\} \geq 0 \end{gathered} \tag{7.25}$$

I can show that:

$$\log\left(\frac{1-\alpha_A-\alpha_x}{1-\alpha_A-\alpha_x-\alpha_{BA}}\frac{1-\alpha_A-\alpha_{BA}}{1-\alpha_A}\right) = \log\left(\frac{1-\frac{\alpha_{BA}}{1-\alpha_A}}{1-\frac{\alpha_{BA}}{1-\alpha_A-\alpha_x}}\right) \geq 0 \tag{7.26}$$

I can thus find a lower bound for the LHS of the inequality (7.25):

$$LHS \geq [-1+\alpha_A]q(S-A, S-A-\{x\}, S-B, S-B-\{x\}) - \alpha_A q(B+\{x\}, B, A+\{x\}, A) + \alpha_A\lambda\log\left(\frac{\alpha_A+\alpha_x}{\alpha_A+\alpha_x+\alpha_{BA}}\frac{\alpha_A+\alpha_{BA}}{\alpha_A}\right) \geq 0 \tag{7.27}$$

Furthermore, I can show that

$$\log\left(\frac{\alpha_A+\alpha_x}{\alpha_A+\alpha_x+\alpha_{BA}}\frac{\alpha_A+\alpha_{BA}}{\alpha_A}\right) = \log\left(\frac{1+\frac{\alpha_{BA}}{\alpha_A}}{1+\frac{\alpha_{BA}}{\alpha_A+\alpha_x}}\right) \geq 0 \tag{7.28}$$

Thus, all I need to impose to guarantee the 3^{rd} and 4^{th} factorizations do not cause any ambiguities, is $q(C, C_1, D, D_1) \leq 0$ for all sets $C_1 \subseteq C, D_1 \subseteq D$ and $D \subseteq C$.

Note 3:: In this case there are two lower bound denominators $\alpha_A+\alpha_{BA}+\alpha_x$ and $1-\alpha_A-\alpha_{BA}$. The first denominator cannot be equal to 0 because $\alpha_x > 0$. If the second denominator is equal to 0, that means that $\alpha_A = 1-\alpha_x \leq alpha_B \leq 1-\alpha_x$ which means that $A = B$ thus inequality (3.8) will definitely hold true.

□

Appendix B: Supplementary Material for Chapter 4

B.1. Theorem III. 1

Proof. For an easier understanding, my proof of the theorem is broken into two sections:

Sufficient Conditions for $\mathbf{F_1(S_j)}$:

Starting for $F_1(S_j)$, by rewriting Eq. (4.15) I will have:

$$\begin{aligned} &\alpha_{BA}[-g(B+\{x\},B)+\lambda\log[\frac{1}{1+\frac{\alpha_x}{\alpha_A+\alpha_{BA}}}]] \\ &+\alpha_x[-g(B+\{x\},A+\{x\})+\lambda\log[\frac{1}{1+\frac{\alpha_{BA}}{\alpha_A+\alpha_x}}]] \\ &+\alpha_A[-q(B+\{x\},B,A+\{x\},A)+\lambda\log\frac{1+\frac{\alpha_{BA}}{\alpha_A}}{1+\frac{\alpha_{BA}}{\alpha_A+\alpha_x}}\geq 0 \end{aligned} \tag{7.29}$$

where I have used the definition of functions f and g.

Since the above inequality needs to hold true for all possible sets of $A \subseteq B$, $x \notin B$, I aim to determine the maximal amount enforced by the above set of inequalities. The first factorization results in two inequalities: (1) $g(C,D) \leq 0$ and (2) $|g(C,D)| \geq \lambda max(\log(1+\frac{\alpha_x}{\alpha_B}))$ for all sets $D \subseteq C$. To find the maximum of such a limit, I need to impose the one item with highest probability to $\{x\}$ and assume the one lowest probability item to set B. Then the above inequality is maximized.

The second factorization results in two inequalities: (1) $g(C,D) \leq 0$ and (2) $|g(C,D)| \geq \lambda max(\log(1+\frac{\alpha_{BA}}{\alpha_A+\alpha_x}))$ for all sets $D \subseteq C$. To find the maximum of such a limit, I need to impose the item with lowest probability to $\{x\}$ and that $\alpha_A = 0$ and then have $\alpha_{BA} = 1-\alpha_x = \alpha_B$.

The third factorization could be simplified. The logarithm argument consists of a nominator greater than denominator thus resulting in the overall logarithm argument to be positive. I thus only need to impose that $q(C, C_1, D, D_1) \leq 0$ for all sets $C_1 \subseteq C, D_1 \subseteq D$ and $D \subseteq C$.

It then follows that if the probability of each item is sorted in a decreasing manner as $\pi_1, ..., \pi_M$, I can write the set of 3 sufficient conditions for submodularity of set function $F_1(S_j), j = 1, ..., N$ as (1) $g(C, D) \leq 0$ (2) $q(C, C_1, D, D_1) \leq 0$ (3) $|g(C, D)| \geq \lambda \log\left[max(1 + \frac{\pi_1}{\pi_M}, 1 + \frac{1-\pi_M}{\pi_M})\right] = \lambda \log\left(\frac{1}{\pi_M}\right)$ for all sets $C_1 \subseteq C, D_1 \subseteq D$ and $D \subseteq C$.

Sufficient Conditions for $\mathbf{F_2(S_j)}$: I now follow the same method for $F_2(S_j), j = 1, ..., N$ by rewriting Eq. (4.15):

$$
\begin{aligned}
&\alpha_{BA}[-g(S - B, S - B - \{x\}) - \lambda \log(1 + \frac{\alpha_x}{1 - \alpha_A - \alpha_{BA} - \alpha_x})] \\
&+\alpha_x[-g(S - A - \{x\}, S - B - \{x\}) - \lambda \log(1 + \frac{\alpha_{BA}}{1 - \alpha_A - \alpha_x - \alpha_{BA}})] \\
&+\alpha_A[q(S - A, S - A - \{x\}, S - B, S - B - \{x\}) \\
&\quad +\lambda \log(\frac{1 - \alpha_A}{1 - \alpha_A - \alpha_x}\frac{1 - \alpha_A - \alpha_{BA} - \alpha_x}{1 - \alpha_A - \alpha_{BA}})] \\
&+1[-q(S - A, S - A - \{x\}, S - B, S - B - \{x\}) \\
&\quad +\lambda \log(\frac{1 - \alpha_A - \alpha_x}{1 - \alpha_A}\frac{1 - \alpha_A - \alpha_{BA}}{1 - \alpha_A - \alpha_{BA} - \alpha_x})] \geq 0
\end{aligned} \tag{7.30}
$$

where I have used the definition of functions f and g.

Since the above inequality needs to hold true for all possible sets of $A \subseteq B$, $x \notin B$, I aim to determine the maximal amount enforced by the above set of inequalities. The first factorization results in two inequalities: (1) $g(C, D) \leq 0$ and (2) $|g(C, D)| \geq \lambda \log(1 + \frac{\alpha_x}{1-\alpha_B-\alpha_x})$ for all sets $D \subseteq C$. To find the maximum of such a limit, I need to impose the one item with highest probability to $\{x\}$ and assume that α_B is maximal while still less than $1 - \alpha_x$. This limit is imposed so that the denominator is not equal to 0. Then the lower bound for $g(C, D)$ will be equal to $\lambda \log(1 + \frac{\pi_1}{\pi_M})$.

The second factorization results in two inequalities: (1) $g(C, D) \leq 0$ and (2) $|g(C, D)| \geq \lambda max(\log(1 + \frac{\alpha_{BA}}{1-\alpha_B-\alpha_x}))$ for all sets $D \subseteq C$. To find the maximum of such a limit, I once again need to impose the one item with highest probability to $\{x\}$ and assume that $\alpha_A = 0$ and α_B is maximal while still less than $1 - \alpha_x$. This limit is once again imposed so that the denominator is not equal to 0. The secondary lower bound for $g(C, D)$ will be equal to $\lambda \log(1 + \frac{1-\pi_M-\pi_{M-1}}{\pi_M})$.

The third and forth factorization could be simplified. I could write them as

$$\begin{aligned}
&[-1 + \alpha_A]\{q(S - A, S - A - \{x\}, S - B, S - B - \{x\}) \\
&+\lambda \log \frac{1 - \alpha_A - \alpha_x}{1 - \alpha_A} \frac{1 - \alpha_A - \alpha_{BA}}{1 - \alpha_A - \alpha_{BA} - \alpha_x}\}+ \\
&+\alpha_A \lambda \log(1) \geq 0
\end{aligned} \tag{7.31}$$

Furthermore, the logarithmic argument is always positive seeing as how the nominator of the fraction inside it is greater than the denominator. It once more thus follows that I merely need to impose $q(C, C_1, D, D_1) \leq 0$ for all sets $C_1 \subseteq C, D_1 \subseteq D$ and $D \subseteq C$ to help above inequality hold true.

Thus 3 sufficient conditions for submodularity of $F_2(S_j), j = 1, ..., N$ are developed: (1) $g(C, D) \leq 0$ (2) $q(C, C_1, D, D_1) \leq 0$ (3) $|g(C, D)| \geq \lambda \log(\frac{1}{\pi_M})$ for all sets $C_1 \subseteq C, D_1 \subseteq D$ and $D \subseteq C$.

Finally, the intersection of the two sets of sufficient conditions for submodularity of $F_1(S_j)$ and $F_2(S_j)$ gives us the sufficient conditions for submodularity of $F(S_j)$ which turns out to be the same as either of theirs.

$\square$

References

[1] Angwin, Julia, Jeff Larson, Surya Mattu, and Lauren Kirchner. "Machine bias. ProPublica." See https://www. propublica. org/article/machine-bias-risk-assessments-in-criminal-sentencing (2016).

[2] Kay, Matthew, Cynthia Matuszek, and Sean A. Munson. "Unequal representation and gender stereotypes in image search results for occupations." In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, pp. 3819-3828. 2015.

[3] Wu, Shuhang, Shuangqing Wei, Yue Wang, Ramachandran Vaidyanathan, and Jian Yuan. "Partition information and its transmission over boolean multi-access channels." IEEE Transactions on Information Theory 61, no. 2 (2014): 1010-1027.

[4] Alon, Noga, and Vera Asodi. "Learning a hidden subgraph." SIAM Journal on Discrete Mathematics 18, no. 4 (2005): 697-712.

[5] Bayat, Farhang, and Shuangqing Wei. "Sequential detection of disjoint subgraphs over boolean mac channels: A probabilistic approach." In 2016 IEEE Globecom Workshops (GC Wkshps), pp. 1-6. IEEE, 2016.

[6] Bayat, Farhang, and Shuangqing Wei. "Non-adaptive sequential detection of active edge-wise disjoint subgraphs under privacy constraints." IEEE Transactions on Information Forensics and Security 13, no. 7 (2018): 1615-1625.

[7] Bayat, Farhang, and Shuangqing Wei. "Partition of Random Items: Trade-off between Binning Utility and Meta Information Leakage." In 2018 25th International Conference on Telecommunications (ICT), pp. 356-360. IEEE, 2018.

[8] Bayat, Farhang and Shuangqing Wei, "Partition of Random Items: Tradeoff between Binning Utility, Meta Information Leakage and Hypotheses Distinguishability", See https://archive.org/details/fullpaper3 2019.

[9] Tishby, Naftali, Fernando C. Pereira, and William Bialek. "The information bottleneck method." arXiv preprint physics/0004057 (2000).

[10] Blahut, Richard. "Computation of channel capacity and rate-distortion functions." IEEE transactions on Information Theory 18, no. 4 (1972): 460-473.

[11] Arimoto, Suguru. "An algorithm for computing the capacity of arbitrary discrete memoryless channels." IEEE Transactions on Information Theory 18, no. 1 (1972): 14-20.

[12] Bertsekas, Dimitri P. "Nonlinear programming. athena scientific." Belmont, MA (1999).

[13] Hong, Mingyi, Zhi-Quan Luo, and Meisam Razaviyayn. "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems." SIAM Journal on Optimization 26, no. 1 (2016): 337-364.

[14] Bayat, Farhang, and Shuangqing Wei. "Information Bottleneck Problem Revisited." In 2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton), pp. 40-47. IEEE, 2019.

[15] Ghassami, AmirEmad, Sajad Khodadadian, and Negar Kiyavash. "Fairness in supervised learning: An information theoretic approach." In 2018 IEEE International Symposium on Information Theory (ISIT), pp. 176-180. IEEE, 2018.

[16] Diaz, Mario, Hao Wang, Flavio P. Calmon, and Lalitha Sankar. "On the robustness of information-theoretic privacy measures and mechanisms." IEEE Transactions on Information Theory (2019).

[17] Nowak, Robert. "Noisy generalized binary search." In Advances in neural information processing systems, pp. 1366-1374. 2009.

[18] Naghshvar, Mohammad, Tara Javidi, and Kamalika Chaudhuri. "Bayesian active learning with non-persistent noise." IEEE Transactions on Information Theory 61, no. 7 (2015): 4080-4098.

[19] Wu, Shuhang, Shuangqing Wei, Yue Wang, Ramachandran Vaidyanathan, and Jian Yuan. "Asymptotic error free partitioning over noisy Boolean multiaccess channels." IEEE Transactions on Information Theory 61, no. 11 (2015): 6168-6181.

[20] Atia, George K., and Venkatesh Saligrama. "Boolean compressed sensing and noisy group testing." IEEE Transactions on Information Theory 58, no. 3 (2012): 1880-1901.

[21] Karbasi, Amin, and Morteza Zadimoghaddam. "Sequential group testing with graph constraints." In 2012 IEEE information theory workshop, pp. 292-296. Ieee, 2012.

[22] Cheraghchi, Mahdi, Amin Karbasi, Soheil Mohajer, and Venkatesh Saligrama. "Graph-constrained group testing." IEEE Transactions on Information Theory 58, no. 1 (2012): 248-262.

[23] Singh, Navraj, Benjamin A. Miller, Nadya T. Bliss, and Patrick J. Wolfe. "Anomalous subgraph detection via sparse principal component analysis." In 2011 IEEE Statistical Signal Processing Workshop (SSP), pp. 485-488. IEEE, 2011.

[24] Fortunato, Santo. "Community detection in graphs." Physics reports 486, no. 3-5 (2010): 75-174.

[25] Ahn, Yong-Yeol, James P. Bagrow, and Sune Lehmann. "Link communities reveal multiscale complexity in networks." nature 466, no. 7307 (2010): 761-764.

[26] Evans, T. S., and Renaud Lambiotte. "Line graphs, link partitions, and overlapping communities." Physical Review E 80, no. 1 (2009): 016105.

[27] Galbrun, Esther, Aristides Gionis, and Nikolaj Tatti. "Overlapping community detection in labeled graphs." Data Mining and Knowledge Discovery 28, no. 5-6 (2014): 1586-1610.

[28] Newman, Mark EJ, and Aaron Clauset. "Structure and inference in annotated networks." Nature communications 7, no. 1 (2016): 1-11.

[29] Gadde, Akshay, Eyal En Gad, Salman Avestimehr, and Antonio Ortega. "Active learning for community detection in stochastic block models." In 2016 IEEE International Symposium on Information Theory (ISIT), pp. 1889-1893. IEEE, 2016.

[30] Campan, Alina, Yasmeen Alufaisan, Traian Marius Truta, and T. Richardson. "Preserving Communities in Anonymized Social Networks." Trans. Data Privacy 8, no. 1 (2015): 55-87.

[31] Nowak, Robert. "Generalized binary search." In 2008 46th Annual Allerton Conference on Communication, Control, and Computing, pp. 568-574. IEEE, 2008.

[32] Nowak, Robert D. "The geometry of generalized binary search." IEEE Transactions on Information Theory 57, no. 12 (2011): 7893-7906.

[33] Ganapathy, S., and V. Rajaraman. "Information theory applied to the conversion of decision tables to computer programs." Communications of the ACM 16, no. 9 (1973): 532-539.

[34] Hartmann, C., Pramod Varshney, Kishan Mehrotra, and C. Gerberich. "Application of information theory to the construction of efficient decision trees." IEEE Transactions on information theory 28, no. 4 (1982): 565-577.

[35] Wu, Shuhang, Shuangqing Wei, Yue Wang, Ramachandran Vaidyanathan, Jian Yuan, and Xiqin Wang. "Detection of graph structures via communications over a multiaccess Boolean channel." In 2015 IEEE International Symposium on Information Theory (ISIT), pp. 2553-2557. IEEE, 2015.

[36] Wu, Shuhang, Shuangqing Wei, Yue Wang, Ramachandran Vaidyanathan, and Jian Yuan. "Achievable partition information rate over noisy multi-access Boolean channel." In 2014 IEEE International Symposium on Information Theory, pp. 1206-1210. IEEE, 2014.

[37] Sobel, Milton, and Phyllis A. Groll. "Group testing to eliminate efficiently all defectives in a binomial sample." Bell System Technical Journal 38, no. 5 (1959): 1179-1252.

[38] Chen, Yuxin, Changho Suh, and Andrea J. Goldsmith. "Information recovery from pairwise measurements." IEEE Transactions on Information Theory 62, no. 10 (2016): 5881-5905.

[39] Zheleva, Elena, and Lise Getoor. "Preserving the privacy of sensitive relationships in graph data." In International Workshop on Privacy, Security, and Trust in KDD, pp. 153-171. Springer, Berlin, Heidelberg, 2007.

[40] Laforet, Fabian, Erik Buchmann, and Klemens Böhm. "Individual privacy constraints on time-series data." Information Systems 54 (2015): 74-91.

[41] Ziegeldorf, Jan Henrik, Oscar Garcia Morchon, and Klaus Wehrle. "Privacy in the Internet of Things: threats and challenges." Security and Communication Networks 7, no. 12 (2014): 2728-2742.

[42] Santiago, Richard, and F. Bruce Shepherd. "Multi-agent and multivariate submodular optimization." CoRR (2016).

[43] Liao, Jiachun, Lalitha Sankar, Vincent YF Tan, and Flavio P. Calmon. "Hypothesis testing in the high privacy limit." In 2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton), pp. 649-656. IEEE, 2016.

[44] Johnstone, David, and Dennis Lindley. "Mean–variance and expected utility: The Borch paradox." Statistical Science 28, no. 2 (2013): 223-237.

[45] Issa, Ibrahim, Aaron B. Wagner, and Sudeep Kamath. "An operational approach to information leakage." IEEE Transactions on Information Theory (2019).

[46] Dwork, Cynthia. "Differential privacy: A survey of results." In International conference on theory and applications of models of computation, pp. 1-19. Springer, Berlin, Heidelberg, 2008.

[47] Liao, Jiachun, Lalitha Sankar, Vincent YF Tan, and Flavio du Pin Calmon. "Hypothesis testing under mutual information privacy constraints in the high privacy regime." IEEE Transactions on Information Forensics and Security 13, no. 4 (2017): 1058-1071.

[48] Wang, Yu, Wotao Yin, and Jinshan Zeng. "Global convergence of ADMM in nonconvex nonsmooth optimization." Journal of Scientific Computing 78, no. 1 (2019): 29-63.

[49] Bayat, Farhang, and Shuangqing Wei. "Partition of Random Items: Tradeoff between Binning Utility and Meta Information Leakage." See https://archive.org/details/ICTTechnicalReport (2019).

[50] Diestel, Reinhard. "Graph theory 3rd ed." Graduate texts in mathematics 173 (2005).

[51] Cook, William J., W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver. "Combinatorial optimization." Oberwolfach Reports 5, no. 4 (2009): 2875-2942.

[52] Feldman, Moran, Joseph Naor, and Roy Schwartz. "A unified continuous greedy algorithm for submodular maximization." In 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science, pp. 570-579. IEEE, 2011.

[53] Ene, Alina, and Huy L. Nguyen. "Constrained submodular maximization: Beyond 1/e." In 2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS), pp. 248-257. IEEE, 2016.

[54] Feldman, Moran. "Maximizing symmetric submodular functions." ACM Transactions on Algorithms (TALG) 13, no. 3 (2017): 1-36.

[55] Resende, Mauricio GC, and Celso C. Ribeiro. "Greedy randomized adaptive search procedures." In Handbook of metaheuristics, pp. 219-249. Springer, Boston, MA, 2003.

[56] Cover, Thomas M., and Joy A. Thomas. "Wiley series in telecommunications and signal processing." In Elements of information theory. Wiley-Interscience, 2006.

[57] Kullback, Solomon, and Richard A. Leibler. "On information and sufficiency." The annals of mathematical statistics 22, no. 1 (1951): 79-86.

[58] Krause, Andreas, and Daniel Golovin. "Submodular function maximization." (2014): 71-104.

[59] Bayat, Farhang, and Shuangqing Wei. "Information Bottleneck Revisited." See https://archive.org/details/isit15 (2019).

[60] Boyd, Stephen, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. "Distributed optimization and statistical learning via the alternating direction method of multipliers." Foundations and Trends® in Machine learning 3, no. 1 (2011): 1-122.

[61] Iutzeler, Franck, Pascal Bianchi, Philippe Ciblat, and Walid Hachem. "Asynchronous distributed optimization using a randomized alternating direction method of multipliers." In 52nd IEEE conference on decision and control, pp. 3671-3676. IEEE, 2013.

[62] Bianchi, Pascal, Walid Hachem, and Franck Iutzeler. "A coordinate descent primal-dual algorithm and application to distributed asynchronous optimization." IEEE Transactions on Automatic Control 61, no. 10 (2015): 2947-2957.

[63] Broward County Clerk's Office, Broward County Sherrif's Office, Florida Department of Corrections, ProPublica, See https://www.propublica.org/datastore/dataset/compas-recidivism-risk-score-data-and-analysis March 2020.

[64] Liao, Jiachun, Oliver Kosut, Lalitha Sankar, and Flavio P. Calmon. "A tunable measure for information leakage." In 2018 IEEE International Symposium on Information Theory (ISIT), pp. 701-705. IEEE, 2018.

[65] Arimoto, S. "Information measures and capacity of order α for discrete memoryless channels." Topics in information theory (1977).

[66] Fehr, Serge, and Stefan Berens. "On the conditional Rényi entropy." IEEE Transactions on Information Theory 60, no. 11 (2014): 6801-6810.

[67] Rebollo-Monedero, David, Jordi Forne, and Josep Domingo-Ferrer. "From t-closeness-like privacy to postrandomization via information theory." IEEE Transactions on Knowledge and Data Engineering 22, no. 11 (2009): 1623-1636.

[68] Sankar, Lalitha, S. Raj Rajagopalan, and H. Vincent Poor. "Utility-privacy tradeoffs in databases: An information-theoretic approach." IEEE Transactions on Information Forensics and Security 8, no. 6 (2013): 838-852.

[69] Verdú, Sergio. "α-mutual information." In 2015 Information Theory and Applications Workshop (ITA), pp. 1-6. IEEE, 2015.

Vita

Farhang Bayat was born on December 15 1990, in Tehran, Iran. He received the B.Sc. and M.Sc. degrees in electrical engineering from Amirkabir University of Technology, Tehran, Iran, in 2012 and 2014, respectively and is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Louisiana State University, Baton Rouge. His research interests include information theory and its applications in graphical models and deep learning systems.