Data Reduction Using EM Algorithm with Deliberately Introduced Missingness

Atanu Kumar Ghosh

INDIAN STATISTICAL INSTITUTE,

KOLKATA

2022

Data Reduction Using EM Algorithm with Deliberately Introduced Missingness

ATANU KUMAR GHOSH

Thesis Advisor: Dr. Arnab Chakraborty

Thesis submitted to the Indian Statistical Institute

in partial fulfillment of the requirements

for the award of the degree of

Doctor of Philosophy.

2022

INDIAN STATISTICAL INSTITUTE

203, B.T. Road, Kolkata, India

Dedicated to my family members.

Acknowledgment

I shall take this opportunity to acknowledge the overall collective support which I received from different persons on various occasions to pursue my research work.

First, I would like to express my sincere gratitude to my supervisor Dr. Arnab Chakraborty. Without his constant academic guidance, this thesis would not appear in its present shape. His academic excellence as well as continuous effort helped me to learn and construct my research findings. His constant cooperation and careful suggestions have always helped me to improve upon my work.

Besides my advisor, I would like to thank Prof. Debasis Sengupta, the Dean of Studies at Indian Statistical Institute for providing me this opportunity to submit my thesis. I am indebted to all my teachers at Indian Statistical Institute, Kolkata, from whom I learned a lot in the form of various academic courses, during my tenure as a Ph.D scholar. I would also like to thank Prof. Anup Dewanji, Prof. Tapas Samanta, Prof. Bimal Kumar Roy, Prof. Subhomoy Moitra, Late Prof. Sourav Ghosh, Prof. Sourav Bhattacharya for their immense support in different issues. I am thankful to the Indian Statistical Institute for providing me with excellent infrastructure to carry out my research work. I would also like to thank various administrative departments of the Indian Statistical Institute for helping me to resolve various administrative matters.

Outside ISI, I would like to express my gratitude towards Presidency University, Kolkata for allowing me to continue my research work. I would like to thank Prof. Biswajit Roy and my other colleagues at the Department of Statistics, Presidency University for their support and encouragement.

I am grateful to my mother Mrs. Sikha Ghosh, my father Mr. Asim Kumar Ghosh and my wife Mrs. Soumana Dey for their immeasurable love and belief they put on me which helped me to complete my work. I also feel proud to have many friends at ISI who made my journey easy and pleasant during the hard times. Finally I thank the anonymous reviewers for their valuable comments which have significantly improved this work.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Not everything that can be counted counts, and not everything that counts can be counted.

— Albert Einstein

1.1. Data Deluge: The problem with modern data

Statistics is supposed to be a data driven subject, but sometimes many of its theoretical developments struggled to find real life applications. In fact, non-availability of suitable data was a point of concern to the statisticians in those days, when the term “data” primarily meant manually collected data. The number of observations in those data sets typically used to be of the order of a few thousands at the most.

However, the story changed drastically with the advent of modern computers, especially of micro-controllers equipped with sensors. In today’s modern sophisticated data collection mechanism *“Over 2.5 quintillion bytes of data are created every single day, and it’s only going to grow from there”*[1] [2]. Ironically the philosophy of “...everything that can be counted counts” has now become the struggle to count everything that can be counted.

[1] According to the sixth edition of DOMO’s report **[2]**.

[2] There are certain application areas like medical statistics, clinical trials where collecting samples are still expensive and difficult.

Over the last several decades, researchers have contributed significantly to the area of digital data acquisition leading to a massive development of Internet of Things (IoT) devices which has lead to this data boom. According to one projection by IDC [**3**], there will be 41.6 billion IoT devices in operation by 2025, which are expected to generate 79.4 zettabytes of data. However a data set cannot be powerful on its own unless a suitable analysis brings out meaningful conclusions from the raw numbers. Modern data are so voluminous, that it is becoming extremely difficult to store the information, let aside analyzing it. This results in an ever increasing gap between the total information produced and the limited nature of the available storage commonly called *data deluge*. As Baraniuk rightly pointed out, *"more sensor data can lead to less efficient sensor systems"*. In his paper [**8**] he illustrates this with the following two scenarios:

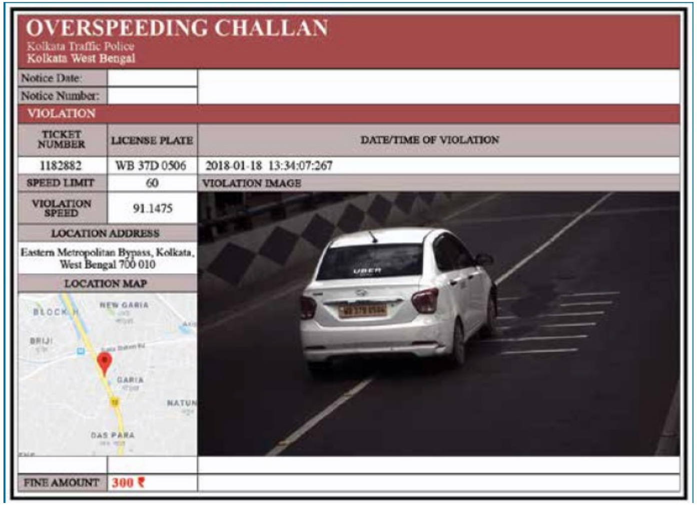
Example 1. (ARGUS-IS) The Defense Advanced Research Projects Agency (DARPA) conducted a project called Autonomous Real-Time Ground Ubiquitous Surveillance Imaging System (ARGUS-IS) in which a 1.8-gigapixel digital camera using hundreds of cell phone camera chips was designed for wide area surveillance. With each camera capable of taking images covering up to 160 square kilometer and capturing 15 frames per second this system is capable of high-resolution monitoring and recording of an entire city. Overall the camera can produce raw data at a rate of 770 gigabits per second (Gbps), which is too much for transmission to the ground station where the maximum possible transmission rate is 274 megabits per second (Mbps).

Example 2. (CMS) The Compact Muon Solenoid (CMS) is a general-purpose detector at the Large Hadron Collider (LHC) which produces data at a rate of 320 terabits per second (Tbps). Such an enormous amount of data is far beyond the capabilities of any processing or storage systems. Hence with the help of a hardware based triage only 800 Gb data per second is selected which are characterized as "interesting" events and subsequently analyzed.

While sophisticated research projects like the ARGUS-IS and CMS have only minimal impact on practical life, the same problem actually plagues even smaller scale data collection scenarios involving digital sensors of evergrowing popularity. The following example illustrates this point:

Example 3. (Traffic monitoring) Kolkata (formerly Calcutta) is one of the most densely populated metropolitan city in India with traffic congested streets. According to one report [**1**], in 2016 alone, there were 13,580 traffic accidents in the roads of Kolkata leading to 11,859 injuries and 6,544 deaths. To reduce overspeeding of vehicles and to prevent traffic signal violations, Kolkata traffic police in association with a Canadian based ITS (Intelligent Traffic Systems) company have installed cameras providing precise images of speeding vehicles' license plates. These images are used by the police to issue immediate challans to the offending drivers. However the total volume of data generated as images and videos is becoming too massive to be stored on any file system.

Figure 1.1.1. Traffic Monitoring



OVERSPEEDING CHALLAN
Kolkata Traffic Police
Kolkata West Bengal

Notice Date:		
Notice Number:		
VIOLATION		
TICKET NUMBER	LICENSE PLATE	DATE/TIME OF VIOLATION
1182882	WB 37D 0506	2018-01-18 13:34:07:267
SPEED LIMIT	60	VIOLATION IMAGE
VIOLATION SPEED	91.1475	
LOCATION ADDRESS		
Eastern Metropolitan Bypass, Kolkata, West Bengal 700 010		
LOCATION MAP		
FINE AMOUNT	300 ₹	

UBER
BLOCK H
NEW GARIA
BRIJI
GARIA
NATUN
DAS PARA

The speed cameras can collect instantaneous information about speed, location and identity of the vehicles along with the time stamp.

There has been a significant amount of research to tackle this problem of data explosion. The management of large scale data has been primarily focusing on increasing the storage capacity of the system. Common examples of this approach include Distributed file handling system and Cloud Computing and Parallel Computing. However these engineering solutions suffer from two major drawbacks:

(1) These are only temporary solutions in the sense that they are in a constant race with the ever growing volume of the data generated. Eventually these solutions will be outrun by the pace of growth in data. In fact these so called solutions merely defer the actual problem for a period of time.

(2) There is a significant waste of resources in terms of storage and management of data. Even if we deploy our best computational efforts, a significant portion of these big data remains

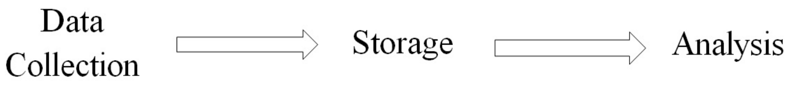
> unused or cannot be subjected to further analysis. According to one report [**24**] in the New York Times, data centres waste 90% of their energy for storing information which will be underutilized.

As an alternative strategy, we propose a new methodology in this thesis. While the conventional engineering solutions aims at increasing the storage and resources, we shall instead assume that it is not possible to increase the primary storage. But thanks to micro-controllers, we can add an extra layer of online computation which can be used suitably to reduce the storage requirement. More specifically the new proposed methodology aims to work with large scale data with limited storage requirements and an additional layer of "light-weight" data processing.

1.2. The new idea of the thesis

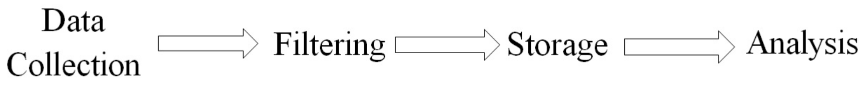
Put naively, the new idea is: "if the volume of incoming data is too big to store or analyze then throw away part of the data...but do so *cleverly* !!!" In face of the lure of ever increasing storage promised by modern hardware, this approach may seem an absurd attempt to ignore technological progress. However, in fact, this approach proposes to utilize another aspect of modern technology that often goes neglected. Micro-controllers are not only good for collecting raw data from sensors, but also capable of performing a light-weight processing of the incoming data on the fly before sending them to permanent storage devices. Figure 1.2.2 shows the steps of our new approach. Indeed it is the

Figure 1.2.1. Steps of Statistical Data Analysis



Statistical data analysis conventionally consists of these three primary steps

Figure 1.2.2. Modified steps of data analysis



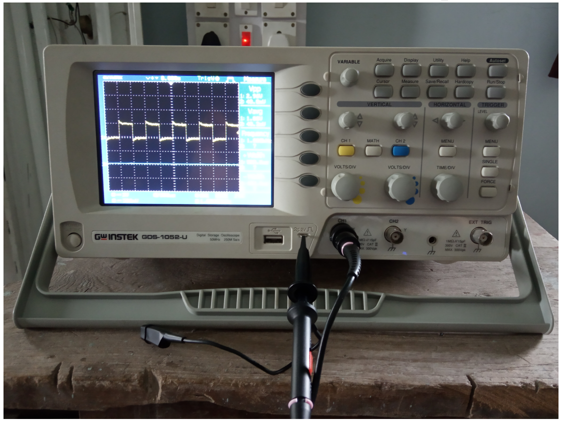
The filtering mechanism adds another step to the conventional steps of data analysis

same as Figure 1.2.1 but for one addition: the "filtering" step which refers to the additional light-weight processing we just talked about.

Preposterous as it may seem to most practicing statisticians, this is nevertheless an idea that has been implemented in certain crude forms earlier also, though these things, to our knowledge, has never been systematically analyzed. We start with one example:

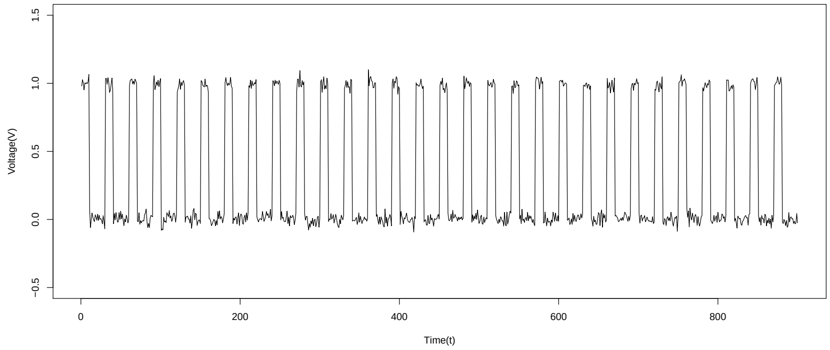
Example 4. (Oscilloscope) Oscilloscope is one of the frequently used instruments in the field of electrical engineering. Reduced to its bare essentials an oscilloscope consists of a moving point plotting out an incoming voltage level on a rectangular screen as a function of time (Figure 1.2.3). The width of the screen typically allows a time span of an order of tens of nano-seconds to micro-seconds. Thus a single swipe only reveals us a small portion of long voltage versus time graph 1.2.4. Once a swipe is complete, the moving dot moves off-screen to the right and must be brought back to the left to start the next swipe.

FIGURE 1.2.3. Oscilloscope



A typical oscilloscope producing waveform from the incoming signals

FIGURE 1.2.4. Instantaneous voltage against time

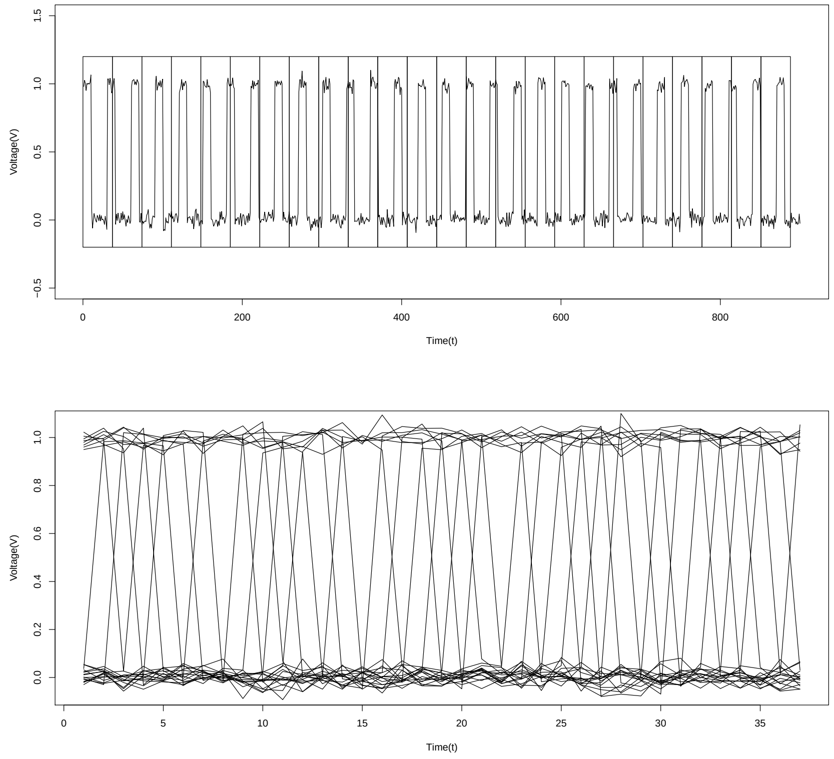


A typical plot of instantaneous voltage against time produces waveform

Typically the voltages measured by an oscilloscope are periodic in nature. The main purpose of the instrument is to detect if the input is indeed periodic and if so, to measure characteristics of the periodic cycles. The jumble shown in Figure 1.2.5 hardly helps to achieve that.

Hence oscilloscopes use a filtering mechanism (commonly called *triggering*): After the completion of a swipe, the next swipe starts

FIGURE 1.2.5. Instantaneous voltages are generally incomprehensible

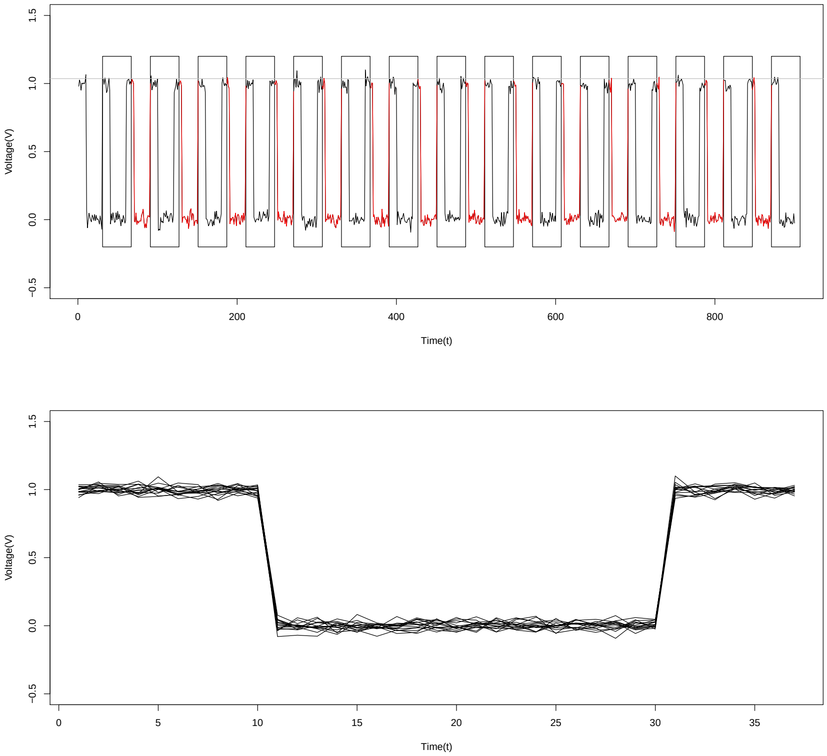


Top panel: The instantaneous voltage plotted with respect to time. The waveform passes the observation window (screen) very fast so that consecutive snapshots on the screen is shown. Bottom panel: The consecutive snapshots are shown simultaneously on the same screen. All these happen in order of nanoseconds making every part of the curve incomprehensible.

only when the input voltage crosses a specified level in a specified direction (upward or downward) as shown in Figure 1.2.6. The resulting display is much more "informative" than Figure 1.2.5, though in reality the former is only a subset of the latter.

This example is a natural illustration of data deluge where "more information" does not imply "more informative". Here cleverly filtering

FIGURE 1.2.6. Trigger function produces comprehensible output

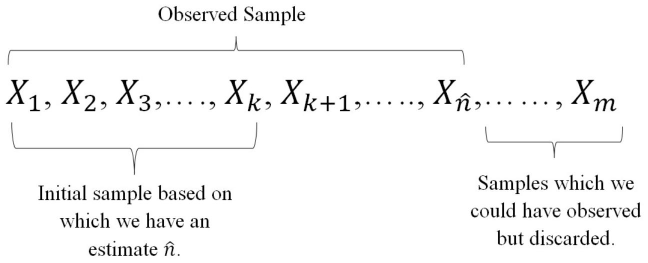


Top panel: The trigger function fixes a threshold value shown by the horizontal line in the plot.The window starts displaying once the input voltage crosses the threshold value until the curve reaches the end of the window and then it discards a part of the curve (shown in red) until the input voltage again touches the threshold. Bottom panel: The trigger function thus produces a stable comprehensible waveform in the screen.

out part of the information makes the data more informative. Here is another example: a more statistical one this time.

EXAMPLE 5. (Pilot Survey) Suppose that in order to estimate the mean μ of a distribution, we can potentially observe a sample of large size m. A standard estimate is the sample mean $\bar{X}$ with standard error

FIGURE 1.2.7. Pilot Survey


Observed Sample
$X_1, X_2, X_3, \ldots, X_k, X_{k+1}, \ldots, X_{\hat{n}}, \ldots, X_m$
Initial sample based on which we have an estimate $\hat{n}$.
Samples which we could have observed but discarded.

We can potentially collect m observations but on the basis of a pilot sample of size k we estimate σ and decide that it is enough to work to with $\hat{n}$ samples to satisfy our error limit. Thus we essentially observe $\hat{n}$ samples and discard the remaining $m - \hat{n}$ observations.

$\frac{\sigma}{\sqrt{m}}$. Let us assume that the sampling procedure is expensive and incurs a fixed cost per observation. Then our aim will be to use only a smaller sample of size $n(\leq m)$, just large enough to achieve a specified level of precision c - that is to keep $\frac{\sigma}{\sqrt{n}} \leq c$. However σ is unknown. In such situations, we often initially collect some samples, say of size k to have an estimate $\hat{\sigma}$ and based on that estimate we decide how many samples we need to observe, say $\hat{n}$, so as to keep the standard error of the estimate $\frac{\hat{\sigma}}{\sqrt{\hat{n}}}$ within the tolerable limit. While this may not be readily recognizable as a data filtering scenario, yet it is one. Here it is like filtering out the $m - \hat{n}$ observations at the very end.

However example 4 further showed that we can extend the idea of filtering observations in the sense that we can ignore observations even from the middle of a stream of observations and yet produce good inference about the underlying population. The two examples we discussed above motivates our proposed methodology which begins with a very simple understanding regarding the nature of data. In most of the applications we shall find that all parts of the data do not bear same significance when it comes to the question of inference. Often we can cleverly filter out a part of the data and can still produce reasonable inferences more easily than with the entire data. This will obviously lead to loss in information, resulting in estimates with less precision, but that loss is more than offset by the gain in computational and storage resources. We shall formalize this idea of filtering later in the thesis. But for now let us focus on two potential application areas which will be helpful while discussing the details of the theory in subsequent chapter. In both of these scenarios, the idea of filtering the data can be obtained easily if we keep in mind the data generation mechanism.

Example 6. (Security Surveillance) Consider a situation where a security camera is deployed in a room to monitor human movement. Modern digital cameras can take frequent snapshots at very small intervals of time. Thus if the camera goes on capturing snapshots throughout the day, the resulting data set will be too big to store. Current engineering solutions tackle this problem with the help of increased storage devices which can store in memory data up to certain days. However if we look at the data we shall find that most of these data will be of no practical use. For example, it may happen that most of the times in day the room remains empty. Hence there will be no significant change in images if we observe them throughout the day. These snapshots taken by the camera is of no practical use. There will be a significant change in images only when someone enters the room and we require that our security camera now captures frequent snapshots to monitor the movement of the person in the room. Clearly we can throw away some of the snapshots of the camera yet getting a good idea about human movement in the room. Hence we want to construct a data collection mechanism where we shall discard a significant portion of the image data but at the same time produce reasonable inferences regrading the underlying population.

Example 7. (Monitoring wind directions) Suppose we monitor the direction of wind at a given place throughout a day in terms of angles (measured in radians). Currently we have sensors which can record the angles at very short intervals of time. However this can produce a huge amount of data throughout the day most of which are constant. This is because wind directions generally remain stable for most part of the day and only fluctuates during a storm or a seldom strong wind. We can definitely use a large storage device and an advanced processor to store all these data, the significant portion of which provides no or little information regarding any storm or major wind fluctuations. Instead if we look at the data generation mechanism and use the fact that wind directions seldom changes throughout the day, then we can discard a large amount of data without affecting the inference too much.

1.3. Organization of the thesis

In Chapter 2, we shall discuss the general idea of filtering mechanism in more mathematical detail. In order to estimate the parameters based on the filtered data, we shall take help of a traditional statistical paradigm known as missing data analysis. The filtering process will be treated as the missing data mechanism which is assumed to be non-ignorable and known. We shall give a brief review of missing data analysis in the context of our problem and more specifically the general idea of EM algorithm is discussed as a standard tool of likelihood based inference in case of missing data problems. Finally we shall have a discussion regarding different aspects in the construction of filtering mechanism.

Chapter 3 discusses the issue of data reduction in case of independent samples. Instead of all the observations we observe only a few chosen linear combinations of them and treat the remaining information as missing. From the observed linear combinations we try to estimate the parameter using EM based technique under the assumption that the parameter is sparse. We shall propose two related methods called ASREM and ESREM for the estimation purpose. The methods developed here are also used for hypothesis testing and construction of confidence interval. Similar data filtering approach already exists in signal sampling paradigm, for example, Compressive Sampling introduced by Donoho [**19**], Candes, Romberg and Tao [**14**]. The methods which we shall propose in this chapter are not claimed to outperform all the available techniques of signal recovery, rather our methods are suggested as an alternative way of data compression using EM algorithm. However, we shall compare our methods to one standard algorithm, *viz.,* robust signal recovery from noisy data using min-ℓ_1 with quadratic constraints. Finally we shall apply one of our methods to a real life dataset.

Chapter 4 deals with the idea of data reduction in case of dependent samples assuming a Markov chain of specified order. Instead of observing all the transitions in a Markov chain we shall record only a few of them and treat the remaining part as missing. The decision about which transitions to be filtered is taken before the observation process starts. Based on the filtered chain we try to estimate the parameters of the Markov model using EM algorithm. In the first half of the chapter we characterize a class of filtering mechanism for which all the parameters remain identifiable. In the later half we explain methods of estimation and testing about the transition probabilities of the Markov chain based on the filtered data. The methods are first developed assuming a simple Markov model with each probability of transition positive, but then generalized for models with structural zeroes in the transition probability matrix. Further extension is also done for multiple Markov chains. The performance of the developed method of estimation is studied using simulated data along with a real life data.

The issue of the construction of filtering mechanism is studied in more detail in Chapter 5. In this chapter we shall continue our discussion on filter matrix introduced in Chapter 4 and consider methods of selection of the optimal filtering matrix. The optimality criteria is defined in terms of the size of the filtered data and as well as the standard error of the estimates. Finally the algorithm developed for the construction of filter matrices will be applied in a real life data set for the purpose of illustration.

CHAPTER 2

On the general idea of filtering

2.1. Introduction

Conventionally multivariate data consist of simultaneous measurements of n individuals on p variables. This is what we have described classically in the form of an $n \times p$ data matrix X. When we say voluminous data, this can actually mean two things:

- an increase in the number of variables (large p).
- an increase in the number of observations (large n).

The former case, conventionally known as high dimensional statistics in the literature, is not our area of concern for this work. Our interest rather revolves round the second case where we have to deal with large number of observations as a result of information explosion. In Chapter 1 we have already discussed that there are available engineering solutions to tackle this problem, all of which focus on upgrading the infrastructure. Instead we introduced an alternative approach which starts by assuming that we have limited storage. As we have mentioned earlier, the central idea of this new approach is to discard a portion of the available data, but wisely. This is accomplished by a process which we termed a filter mechanism in our earlier discussion. The concept of filtering the data has been illustrated repeatedly in different forms in several contexts with the help of examples. In this chapter, we shall provide a formal, unified framework for the concept of filtering mechanism. As we shall see, the unified framework encompasses traditional concepts like grouping data and censoring observations.

It is worthwhile to mention here that the concept of filtering the data is not entirely new and already existed in a different form under the paradigm of data compression. These forms of data compression are significantly different from the idea we are going to introduce here, both in theory and application. However, for the sake of completeness, we feel that there has been a lot of previous work which should be mentioned. The idea of data compression began in 1838 with the invention of Morse code, which uses shorter codewords for the most frequent characters. This concept was used by Claude Shannon [**42**] and Robert Fano [**22**], who devised a systematic way to assign codewords. Huffman [**27**] found an optimal algorithm to implement this idea. In 1977, Abraham Lempel and Jacob Ziv [**50**] suggested an idea of encoding, which in 1984, after the following work by Terry Welch [**48**], became popular as LZW algorithm. The beginning of lossy compression can be attributed to Fourier approximations, where we can decompose any sufficiently smooth function into sums of sine waves with frequencies corresponding to successive integers. Fourier's method is applicable to sounds where after splitting a function into frequencies, we can drop the highest and lowest frequencies, but keep the rest. Wavelet Compression was introduced by Alfred Haar [**25**] and the field emerged rapidly since the 1970s. Depending on the type of data, different compression algorithms have been developed over time. Jayasankar et.al. [**28**] provides a broad survey of the different compression algorithms.

The latter part of this chapter deals separately with two distinct and major components of the proposed methodology :

- designing a suitable filtering mechanism and
- using the filtered data to infer about the underlying population.

While drawing inferences based on the filtered data, we shall treat the data discarded by the filtering mechanism as missing. But unlike conventional missing data analysis, here the missing mechanism will not assumed to be random. In other words, we shall discard a part of the data in the filtering step but during the inference step we shall use the information about how they were discarded. Expectation Maximization (EM) algorithm will be applied for likelihood based inference as a standard tool in incomplete data problems. The filtering mechanism will play a crucial role in the E-step of the algorithm. Regarding construction of the filtering mechanism, several issues are to be considered all of which will be discussed along with examples in Section 6.

2.2. Filtering mechanism

Suppose we have access to samples $X_1, X_2, ..., X_n$ where each $X_i \in \mathbb{R}^p$. Instead of storing all the observations our idea is to store an appropriate many-to-one function of the data and treat the remaining part of the data as missing. We shall call this many-to-one function our filter (F). This function is defined over $\mathcal{X}^n$, where $\mathcal{X}$ is the sample space and n is the sample size. At this point, two important points regarding the filter operator are to be noted:

- Filter operator does not in any way reduce the number of variables, that is, filtering the data is not a dimension reduction technique.
- As we shall see in subsequent sections filtering the data is an online process acting on the stream of incoming observations but conceptually filter operator is defined as functions of the hypothetical complete data.

Let us now understand the concept of filtering with the help of some examples some of which assume independent observations and others assume dependent data setup.

Example 8. In the simplest case F can lead to a proper subset of the original data where we retain only some k observations and discard the remaining ones. Suppose the filtered data are $Y_1, Y_2, ..., Y_k$ where $Y_j = X_{i_j}, j = 1, 2, ..., k$ and $i_j \in \{1, 2, ..., n\}$.

Example 9. F can be k linear combinations of the sample observations like

$$Y_i = \ell_{i1}X_1 + \ell_{i2}X_2 + ... + \ell_{in}X_n, i = 1, 2, ..., k.$$

In this case F can be expressed explicitly in the form of an $k \times n$ matrix with elements ℓ_{ij}.

EXAMPLE 10. Suppose we have the available observations $X_1, X_2, ..., X_n$. Instead of storing all the data we group them into 5 fixed and known categories $C_i, i = 1, 2, ..., 5$ which are mutually exclusive and exhaustive. The observed frequencies $f_i, i = 1, 2, ..., 5$ of the categories are only recorded. Here the filtering operator F takes the form

$$f_i = F(X_1, X_2, ..., X_n) = \sum_{j=1}^{n} I(X_j \in C_i) \text{ for all } i = 1, 2, ..., 5.$$

EXAMPLE 11. The previous example can be extended to the setup of what is called the Tobit model in the literature. Consider a multiple regression with a response variable Y and p covariates $X_1, X_2, ..., X_p$ where the values of Y are grouped into categories but the covariates are completely observed. In particular, this include censoring of the Y values where the positive Y values are retained and the negative values are censored. Here the filtered data are

$$Z_i = Y_i I(Y_i > 0) \text{ for all } i.$$

It is often convenient to think filter operator as a computational algorithm as it is not always convenient to express F explicitly as a mathematical function.

EXAMPLE 12. Consider $n = 100$ observations $X_1, X_2, ..., X_{100}$ from the following autoregressive process

$$X_i = \phi X_{i-1} + \epsilon_i, i = 1, 2, ..., 100$$

where ϵ_i are iid samples from $N(0, \sigma_0^2)$ distribution and σ_0^2 and X_0 are known. Further let us assume $|\phi| < 1$ and ϕ is the parameter of interest. Let us consider three filter operators:

(1) F_1 : We retain only the first 90 samples and discard the last 10 observations.

(2) F_2 : We discard every 10^{th} observation from the beginning and retain the remaining observations.

(3) F_3 : We discard any X_i if $|X_i - X_{i-1}| < c$ for $i = 1, 2, ..., 100$ and some given c and if any X_{i-1} is discarded then X_i is definitely retained.

Technically any sort of collapsing of the data can be treated as filtering mechanism, but for our idea to work we shall be interested in filtering mechanisms which lead to resonable inferences based on the filtered data. Given a filtering mechanism, how do we perform statistical inference based only on the filtered data? As mentioned earlier this requires analysis of missing data. We shall treat the original data as the complete data and the filtered data as the observed data. The following three sections provide a quick review of the various standard concepts in missing data analysis which will be useful for the development of the proposed methodology.

2.3. A Brief Review of Missing Data Analysis

Missing data refers to the information about the population which is missing in the sense that it cannot be either observed or preserved till analysis. Traditionally missingness occurs as a natural phenomenon, the cause of which can be broadly classified into three categories [**36**]:

(1) missingness due to the study participants,
(2) missingness due to the study design,
(3) the interaction of the participants and the study design.

In general, missing data are viewed as a problem which affects our ability to infer about the population. Hence all of the literature in Statistics are dedicated towards removing the missing data hindrance in data analysis. This includes imputation techniques and likelihood based inferences. In all of the classical works, the statistical methodologies proposed can be grouped into following categories [**32**]:

(1) Procedures Based on Completely Recorded Units
(2) Weighting Procedures
(3) Imputation based procedures
(4) Model based procedures

2.3.1. Missing Data mechanisms: The most widely accepted missing data classification system was introduced by Donald Rubin (1976) [**41**] which concerns the relationship between missingness and the values of variables in the data matrix. Specifically Rubin discussed three types of missing data mechanisms:

- missing completely at random (MCAR) where the missing data are unrelated to both the missing responses and the set of observed responses, the observed values are representative of the entire sample without missing values.

- missing at random (MAR) where the missing data depend on the set of observed responses but are unrelated to the missing values.
- missing not at random (MNAR) where the missing data are related to specific missing values.

Thus the three missing data mechanisms can be made precise by formalizing the relationship between the missingness indicator and the data. Consider n simultaneous measurements on p variables as $x_{ij}, i = 1, 2, ..., n, j = 1, 2, ..., p$ collected in a data matrix X. Further let us define the missing data indicator matrix M with elements $m_{ij}, i = 1, 2, ..., n, j = 1, 2, ..., p$ such that

$$m_{ij} = \begin{cases} 1 & \text{if } x_{ij} \text{ is missing} \\ 0 & \text{if } x_{ij} \text{ is observed} \end{cases}.$$

Further let us denote the observed component of X as X_{obs} and the missing component as X_{mis}. Missing data mechanism refers to relation of M with X and can be characterized by the conditional distribution of M given X, say $f(M|X, \theta)$ for some unknown parameter θ. Specifically the missing data mechanism is

- MCAR if $f(M|X, \theta) = f(M|\theta)$ for all X, θ.
- MAR if $f(M|X, \theta) = f(M|X_{obs}, \theta)$ for all X_{mis}, θ.
- MNAR if $f(M|X, \theta)$ is function of X_{mis}.

This distinction of missing data mechanism is further extended in a more recent paper by Mealli and Rubin (2015) **[38]** where the authors have precisely clarified the idea of missingness at random. In particular, there is a distinction between the missing data being missing at random (which is defined by MCAR or MAR as above) and the missingness mechanism which always produces data that are missing at random (which they have referred to as missing always at random). Similarly, we can make distinction between missing not at random (MNAR) defined above and missing not always at random (MNAAR) where the missing mechanism is such that it always produces data which are missing not at random. While the distinction between the missing data mechanisms should be clear by this time, we shall declare that this work focuses on the non-ignorable missing data mechanism only as described in the following section.

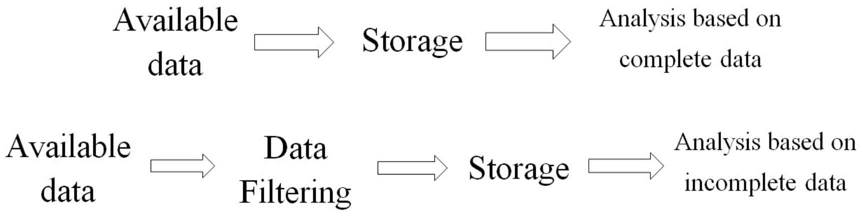
2.4. Bet on non-ignorable missing data mechanism

Traditionally what has been mostly missing from the literature of "missing data analysis" is the study of missing mechanism. However there have been instances of the study of the missing data mechanism in some literature where missingness is introduced intentionally to deal with one or other kind of problems. This includes applications in design of experiments (Hocking & Smith, 1972 [**26**]; Trawinski & Bargmann, 1964 [**47**], Kempthorne, 1952 [**29**]), sequential analysis (Lehman, 1959 [**31**]) and sampling from a finite population (Cochran, 1963 [**16**]). The novelty of this work is that unlike most of the traditional applications where missingness occurs naturally as a problem, here missingness is deliberately introduced to reduce the effective size of the data to be stored. As we have already mentioned earlier, the total data which we observe or could have been observed, if we have enough resources, is the complete data and the data which we actually store after the filtration process is the observed data. The filter operator F therefore plays the role of missing data mechanism M which is assumed to be non-ignorable and completely known. The reasons behind considering the missing data mechanism to be non-ignorable is that we shall exploit the filtering mechanism while estimating our parameters. That is, we shall discard some information from the available data but rather than simply forgetting them, we remember how they were discarded so that we use that piece of information for estimation. We shall see later with the help of examples how discarding the same quantity of data in different ways may lead to parameter estimates of varying accuracy. If the missing data mechanism is ignored during estimation, then the size of the filtered data is all what affects the estimate. However the study of missing data mechanism in our case will justify how to discard observations when we need to. Another aspect we need to mention at this point that the non-ignorable missing data mechanism can be exploited if we perform likelihood based inference for the parameters.

2.5. Likelihood based inference

Model based inference has been an important aspect of missing data analysis in various contexts. In this work we shall be interested in method of estimation based on likelihood function under specific model assumptions.

FIGURE 2.5.1. From complete data analysis to incomplete data analysis


Available data
Storage
Analysis based on complete data
Available data
Data Filtering
Storage
Analysis based on incomplete data

Inference based on filtered data means doing analysis on the basis of incomplete data. Traditionally without the filtering step we can afford analysis based on the complete data.

2.5.1. Likelihood based estimation based on complete data: Following our earlier notation, suppose X denote the complete data that we would receive if there is no filtering mechanism. The likelihood based approach requires that we assume a parametric model, described by the probability distribution $f_\theta(X), \theta \in \Theta$, which generates the data. Then the likelihood of θ is any function $L(\theta|X), \theta \in \Theta$ which is proportional to $f_\theta(X)$. Since this likelihood is based on the complete data we shall denote this as $L_{com}(\theta)$ or the corresponding log-likelihood as $\ell_{com}(\theta) = \log L(\theta|X), \theta \in \Theta$. In case of maximum likelihood estimation we choose the value of $\theta \in \Theta$ which maximizes $L_{com}(\theta)$ or equivalently $\ell_{com}(\theta)$. However due to limitation of storage resources, we need to apply the filtering mechanism, as a result of which the complete data is not available. Hence the maximum likelihood estimate of θ cannot be obtained by minimizing the complete data log-likelihood $\ell_{com}(\theta)$ and we need to rely on the estimation based on the incomplete data.

2.5.2. Likelihood based estimation based on incomplete data: As before let us write $X = (X_{obs}, X_{mis})$, where X_{obs} and X_{mis}

denote the observed and the missing component of X. The joint probability distribution is given by $f_\theta(X)$ which is defined as $f_\theta(X) = f(X|\theta) = f(X_{obs}, X_{mis}|\theta)$. Further suppose M be the missing data indicator. Then the joint distribution of M and X is given by

$$f(X, M|\theta, \psi) = f(X|\theta)f(M|X, \psi), (\theta, \psi) \in \Theta_{\theta,\psi}.$$

The actual observed data is (X_{obs}, M) whose distribution of the observed data can be obtained as

$$f(X_{obs}, M|\theta, \psi) = \int f(X, M|\theta, \psi) dX_{mis}$$

and the likelihood of θ and ψ is any function of θ and ψ proportional to this joint distribution as

$$L(\theta, \psi|X_{obs}, M) \propto f(X_{obs}, M|\theta, \psi), (\theta, \psi) \in \Theta_{\theta,\psi}.$$

We shall call this likelihood the likelihood based on the observed data when the missing mechanism is non-ignorable and denote this as $L_{obs}(\theta, \psi)$ or the corresponding log-likelihood as $\ell_{obs}(\theta, \psi)$. In particular we shall be mainly interested in the case where the missing mechanism is non-ignorable but known because of the filter operator is completely fixed. Hence the observed data log-likelihood reduces to $\ell_{obs}(\theta), \theta \in \Theta$. The maximization of $\ell_{obs}(\theta)$ may sound good in principle except the fact that at times, due to the complicated missing data mechanism imposed by specific filter operators, $\ell_{obs}(\theta)$ cannot be obtained explicitly or even if computed cannot be maximized directly.

EXAMPLE 13. (Example 12 continued) If we consider the filter F_1, then the observed data are $X_1, X_2, ..., X_{90}$ and it is immediate to find the maximum likelihood estimate of ϕ based on the observed data as

$$\hat{\phi} = \frac{\displaystyle\sum_{i=1}^{90} x_i x_{i-1}}{\displaystyle\sum_{i=1}^{90} x_i^2}.$$

If the filter F_2 is employed then the observed likelihood can be found out mathematically but it is difficult to maximize the likelihood with respect to the parameter. On the other hand when we apply the filter F_3, the observed likelihood is very difficult to be computed and maximized. Thus maximizing the observed likelihood can be computationally difficult depending on the filter mechanism we choose.

This problem can be largely tackled by using EM algorithm which provides a general protocol of finding the maximum likelihood estimates and as well as standard errors of the estimates under a wide class of filter.

2.5.3. EM Algorithm: EM algorithm is a standard tool for maximum likelihood estimation in incomplete data problems. The biggest advantage of EM algorithm which we shall be exploiting is: even in situations where there are no actual missing data, the given problem can be reformulated as a missing data problem and can be tackled with the EM algorithm. EM algorithm is an iterative procedure which generalizes the intuitive concept of filling in the missing values and estimating the parameter through the following steps of iteration:

(1) Replace the missing values by their estimated figures,

(2) Use the estimated figures to estimate the parameter,

(3) Re-estimate the missing values based on the new parameter estimate,

(4) Re-estimate the parameter,

and so on. However this is different from standard imputation techniques in the sense that "missing data" here does not necessarily refer to the missing values X_{mis} rather functions of X_{mis} which appear in the complete data log-likelihood $\ell_{com}(\theta)$. Each iteration of EM consists of an E-step (expectation step) and an M-step (maximization step). In particular for non-ignorable missing data mechanism, the E-step finds the conditional expectation of the complete data log-likelihood given the observed data, current estimated parameters and the missing data mechanism. The M-step then maximizes that expected log-likelihood to obtain the parameter estimates in the same way as the ML estimation in case of complete data. Thus in case of non-ignorable missing mechanism the steps of EM algorithm can be expressed as:

(1) start with some initial estimates $(\theta^{(0)}, \psi^{(0)})$ of (θ, ψ).

(2) E-step: at t^{th} iteration, given current estimates $(\theta^{(t)}, \psi^{(t)})$ of (θ, ψ), the E-step calculates

$$Q(\theta, \psi|\theta^{(t)}, \psi^{(t)}) = \int \ell_{comp}(\theta, \psi|X_{obs}, X_{mis}, M) f(X_{mis}|X_{obs}, M; \theta = \theta^{(t)}, \psi = \psi^{(t)}) dX_{mis}$$

where $\ell_{comp}(\theta, \psi | X_{obs}, X_{mis}, M)$ is the complete data log-likelihood and $f(X_{mis}|X_{obs}, M; \theta = \theta^{(t)}, \psi = \psi^{(t)})$ is the conditional distribution of the missing data given the observed data and the missing mechanism M and θ, ψ.

(3) M-step: The M-step maximizes Q to find the estimates of the next iteration as

$$Q(\theta^{(t+1)}, \psi^{(t+1)} | \theta^{(t)}, \psi^{(t)}) \geq Q(\theta, \psi | \theta^{(t)}, \psi^{(t)}) \text{ for all } \theta, \psi.$$

We iterate until convergence. As we have already mentioned in our case the missing mechanism M is non-ignorable but completely known, the parameter ψ can be omitted from the above general description of the algorithm for our purpose. It can be shown under regularity conditions each iteration of this algorithm increases $L(\theta, \psi | X_{obs}, M)$, and under general conditions the algorithm converges to a stationary value of the observed likelihood. The following examples illustrates how EM algorithm provides a convenient way of finding the maximum likelihood estimator based on the filtered data under a wide class of filter mechanism.

Example 14. (Example 12 continued) Consider the filter operator F_3 which stores a observation only when it differs from the previous one by at least an amount of c. Moreover the filtering mechanism does not allow more than one consecutive observation to be discarded simultaneously. As we have already discussed finding the maximum likelihood estimator based on the filtered data is not possible in this case. EM algorithm however can be applied in this case to find the estimate of the parameter ϕ. To begin with, we partition the total data as $X = (x_{mis}, x_{obs})$ and D_{obs} and D_{mis} are sets of the indices of the observed data and the missing data, so that $D_{obs} \cup D_{mis} = \{1, 2, ..., 100\}$. Suppose for all $j = 1, 2, ..., n$ we define the following quantities

$$\alpha_j = \frac{x_j - c - \phi x_j}{\sigma_0}, \beta_j = \frac{x_j + c - \phi x_j}{\sigma_0}.$$

The complete data log-likelihood is given by

$$\ell_{comp}(\phi) = \text{constant} - \frac{1}{2\sigma_0^2}\sum_{i=1}^{n}(x_i - \phi x_{i-1})^2$$

which is a linear function of the statistics $\sum x_{i-1}^2$ and $\sum x_i x_{i-1}$. At the t^{th} iteration, the E-step of the algorithm finds

$$\mathcal{E}_1 = E\Big(\sum_{i=1}^{n} x_{i-1}^2 | x_{obs}, F_3; \phi^{(t)}\Big) \text{ and } \mathcal{E}_2 = E\Big(\sum_{i=1}^{n} x_i x_{i-1} | x_{obs}, F_3; \phi^{(t)}\Big).$$

Now

$$\mathcal{E}_1 = \sum_{j \in D_{obs}} x_{j-1}^2 + \sum_{j \in D_{mis}} E\Big(x_{j-1}^2 | x_{obs}, F_3; \phi^{(t)}\Big)$$

where

$$E\Big(x_j^2 | x_{obs}, F_3; \phi^{(t)}\Big) = E\Big(x_j^2 | x_j \in (x_{j-1} - c, x_{j-1} + c); \phi^{(t)}\Big)$$

$$= \sigma_0^2 + \phi^{(t)2} x_{j-1}^2 + \sigma_0^2 \frac{\alpha_{j-1} f(\alpha_{j-1}) - \beta_{j-1} f(\beta_{j-1})}{\Phi(\alpha_{j-1}) - \Phi(\beta_{j-1})} + 2\sigma_0 \phi^{(t)} x_{j-1} \frac{f(\alpha_{j-1}) - f(\beta_{j-1})}{\Phi(\alpha_{j-1}) - \Phi(\beta_{j-1})}$$

where $f(.)$ and Φ are the p.d.f and c.d.f. of standard normal distribution respectively and α_{j-1} and β_{j-1} are evaluated under the current parameter estimate $\phi^{(t)}$. Similarly we have

$$\mathcal{E}_2 = \sum_{j: j, j-1 \in D_{obs}} x_j x_{j-1} + \sum_{j: j \in D_{mis}, j-1 \in D_{obs}} x_{j-1} E\Big(x_j | x_{obs}, F_3; \phi^{(t)}\Big)$$

$$+ \sum_{j:j\in D_{obs},j-1\in D_{mis}} x_j E\Big(x_{j-1}|x_{obs}, F_3; \phi^{(t)}\Big)$$

where

$$E\Big(x_j|x_{obs}, F_3, \phi^{(t)}\Big) = E\Big(x_j|x_j \in (x_{j-1} - c, x_{j-1} + c); \phi^{(t)}\Big)$$

$$= \phi^{(t)} x_{j-1} + \sigma_0 \frac{f(\alpha_{j-1}) - f(\beta_{j-1})}{\Phi(\alpha_{j-1}) - \Phi(\beta_{j-1})}.$$

The M-step maximizes the expected log-likelihood and the estimate is found out from the MLE based on the complete data with the observations replaced by their expectations as

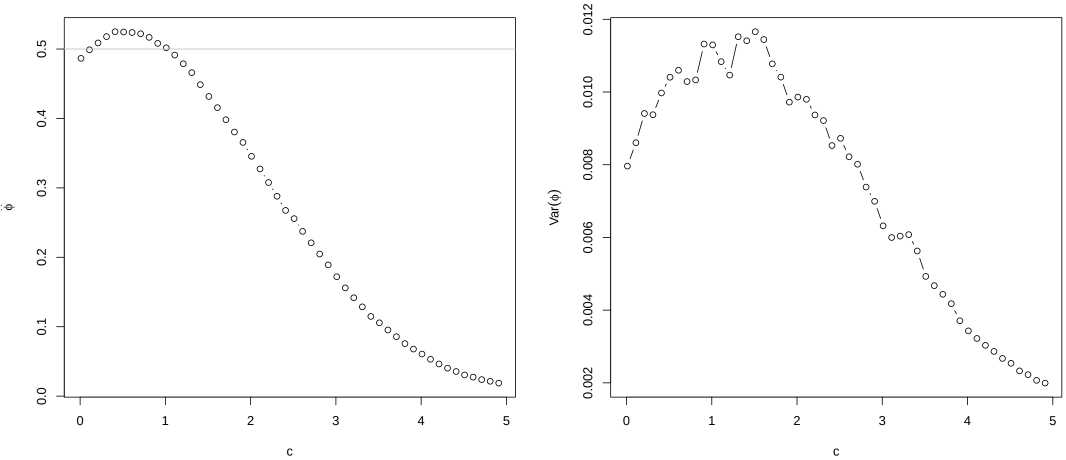
$$\phi^{(t+1)} = \frac{\mathcal{E}_1}{\mathcal{E}_2}.$$

We iterate until convergence. The quality of the estimate varies with the nature and amount of the filtered data which in turn are governed by the choice of c. In order to have an idea of how the choice of c affects the parameter estimate, we perform a simulation study with $X_0 = 0$,$\phi = 0.5$,$\sigma_0 = 1$ and different choices of c varying between 0.01 to 5. Figure 2.5.2 shows the plot of $\hat{\phi}$ and $\widehat{Var}(\hat{\phi})$ for different values of c. We find that the choice of c is crucial in getting a good estimate. Clearly a very large value of c produces bad estimates.

The following two examples have been adapted from [**32**] and here we have cast them into our filtering paradigm.

Example 15. (Effect of grouping) Suppose have random samples $X_1, X_2, ..., X_n$ from an exponential distribution with unknown mean θ.

FIGURE 2.5.2. Choice of filtering mechanism affects the quality of estimate



The plots are results of a simulation study for the model in Example 12 with $X_0 = 0$, $\phi = 0.5$, $\sigma_0 = 1$ and F_3 as the filtering mechanism. For each value of c in [0.01,5] we plot the parameter estimate $\hat{\phi}$ and $\widehat{Var}(\hat{\phi})$ as a result of 1000 simulations.

Consider a filter operator F which groups the observations into k mutually exclusive and exhaustive classes. More specifically the filtered data are $k(< n)$ values $f_1, f_2, ..., f_k$ where f_i is the number of observations belonging to the i^{th} class $[a_i, b_i)$ where $a_1 = 0$ and $b_k = \infty$. The complete data log-likelihood is a linear function of $\sum_{i=1}^{n} x_i$. Hence in the E-step of the algorithm at t^{th} iteration, we find

$$E\Big(\sum_{i=1}^{n} x_i | f_1, f_2, ..., f_k, F; \theta^{(t)}\Big) = \sum_{j=1}^{k} f_j \hat{x}_j^{(t)}$$

where

$$\hat{x}_j^{(t)} = E\Big(x | a_j \leq x < b_j; \theta^{(t)}\Big)$$

$$= \int_{a_j}^{b_j} y \exp\Big(-\frac{y}{\theta^{(t)}}\Big) dy / \int_{a_j}^{b_j} \exp\Big(-\frac{y}{\theta^{(t)}}\Big) dy$$

$$= \theta^{(t)} + \frac{b_j e^{-b_j/\theta^{(t)}} - a_j e^{-a_j/\theta^{(t)}}}{e^{-b_j/\theta^{(t)}} - e^{-a_j/\theta^{(t)}}}.$$

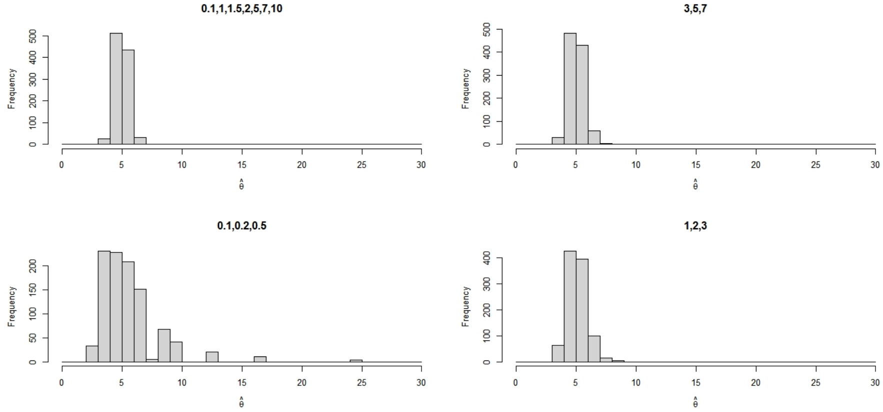
Finally at the M-step of the algorithm we replace the observations in the expression of MLE based on complete data with their expected value and get

$$\hat{\theta}^{(t+1)} = \frac{1}{n}\sum_{j=1}^{k} f_j \hat{x}_j^{(t)}.$$

The missing data mechanism directed by the filter operator in this example is non-ignorable and completely known assuming the choice of a_i and b_i are fixed in advance. However the size of the filtered data depends on the filter operator which in turn is governed by the choice of the number of groups k. The important point is that the variance of the maximum likelihood estimator varies with the number of groups as well as the selection of groups as depicted in Figure 2.5.3. This is due to the information in the observed likelihood which is directed by the filter operator. Hence not only the size of the filtered data but also the type of filter matters - it is important to decide how we group the data.

These examples carry an important message: We may throw away some part of data without significantly hurting the quality of inference. This means it is important to consider how to throw away data when we must, due to the limitation of resources. Further this is exactly why the deliberate introduction of missingness and the non-ignorability of the missing mechanism are so crucial as we mentioned earlier.

FIGURE 2.5.3. Effect of grouping on the variance of the estimator



The figure shows four histograms of the parameter estimates in 1000 simulations corresponding to different grouping (class boundaries except $-\infty, \infty$ are indicated on the top of each figure)

EXAMPLE 16. (Tobit Model Continued) We are considering a regression of Y on p covariates $X_1, X_2, ..., X_p$ where the covariates are fully observed but in case of the response variable we use

$$z_i = y_i I(y_i > 0) \text{ for all } i.$$

Further we assume a multiple linear regression model

$$Y = \beta_0 + \sum \beta_k X_k + \epsilon$$

where $E(\epsilon) = 0$ and $Var(\epsilon) = \sigma^2$. The parameter vector here is $\theta = (\beta_0, \beta_1, ..., \beta_p, \sigma^2)$. The complete data log-likelihood is a linear function of $\sum y_i, \sum y_i^2$ and $\sum y_i x_{ki}, k = 1, 2, ..., p$. Hence after t iterations, at the E-step we calculate

$$E\Big(\sum y_i|Y_{obs}, F, \theta = \theta^{(t)}\Big) = \sum_{i:y_i>0} y_i + \sum_{i:y_i<0} \hat{y}_i^{(t)}$$

$$E\Big(\sum y_i x_{ki}|Y_{obs}, F, \theta = \theta^{(t)}\Big) = \sum_{i:y_i>0} y_i x_{ki} + \sum_{i:y_i<0} \hat{y}_i^{(t)} x_{ki}, k = 1, 2, ...p$$

$$E\Big(\sum y_i^2|Y_{obs}, F, \theta = \theta^{(t)}\Big) = \sum_{i:y_i>0} y_i^2 + \sum_{i:y_i<0} (\hat{y}_i^{(t)2} + \hat{s}_i^{(t)2})$$

where

$$\hat{y}_i^{(t)} = E(y_i|\theta^{(t)}, y_i \leq 0, x_i)$$
$$= \beta_0^{(t)} + \sum \beta_k^{(t)} x_{ki} - \sigma^{(t)} \lambda\Bigg(-\frac{\beta_0^{(t)} + \sum \beta_k^{(t)} x_{ki}}{\sigma^{(t)}}\Bigg)$$

where $\lambda(z) = \frac{\phi(z)}{\Phi(z)}$, ϕ and Φ being the p.d.f. and c.d.f. of $N(0, 1)$ distribution respectively and

$$\hat{s}_i^{(t)2} = \sigma^{(t)2}\Bigg(1 - \delta_i^{(t)}\Big(\delta_i^{(t)} + \frac{\beta_0^{(t)} + \sum \beta_k^{(t)} x_{ki}}{\sigma^{(t)}}\Big)\Bigg)$$

where $\delta_i^{(t)} = \lambda\Bigg(-\frac{\beta_0^{(t)} + \sum \beta_k^{(t)} x_{ki}}{\sigma^{(t)}}\Bigg)$. Further under normality assumption, the maximum likelihood estimator of $\beta_i's$ are same as the ordinary least square estimator. Hence at the M-step of the iteration we perform a ordinary least square regression with $\sum y_i, \sum y_i^2$ and $\sum y_i x_{ki}, k = 1, 2, ..., p$ replaced by their expected values.

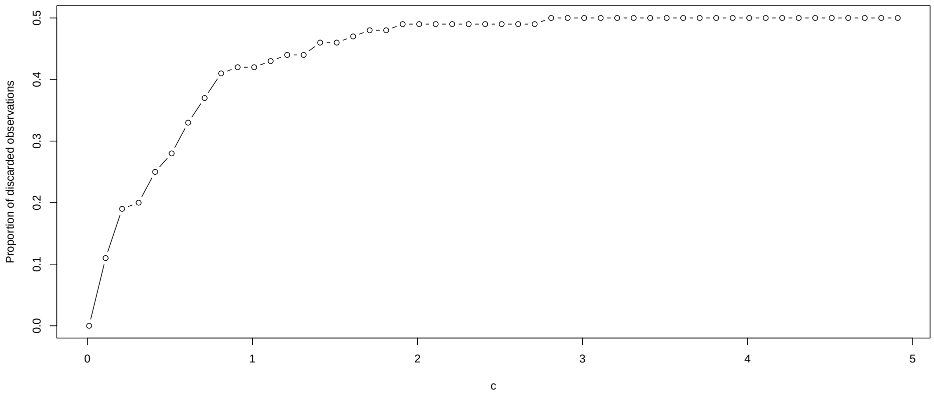
2.6. Constructing the filter operator

So far we have discussed how we can employ the EM algorithm and the general concept of likelihood based inference on filtered data in order to estimate the parameters of the underlying population. All these estimation procedures assume that the initial filtering process is fixed, non-adaptive and completely known in advance. The next important part of the proposed methodology is the construction of the filtering mechanism. Apparently how we should filter the data depends on the available data as well as our available storage. But at the very first place, while designing any filtering mechanism we need to keep in mind the issue of identifiability of the parameters. If we throw away too much data then the parameters of the underlying population may not remain identifiable. In Example 15, the parameter is not identifiable if we choose $k = 1, a_1 = 0, b_1 = \infty$. In general the issue of identifiability depends on the underlying population and the type of filter we are using. Hence we need to ensure that our filter preserves identifiability of the parameters subject to the restriction that the choice of the filter is governed by:

- how much we store
- what we store.

The first choice can be easily make from the size of the available storage. The important thing here is the size of the filtered data may be dictated either explicitly or implicitly by the filtering mechanism. More specifically there can be a filtering mechanism which has one or more tuning parameters which control the size of the filtered data. The choice of

FIGURE 2.6.1. Plot of proportion of discarded data with the choice of c



The figure shows the proportion of discarded observations for different choices of c.

filtering mechanism in such cases boils down to the appropriate choice of the tuning parameter.

EXAMPLE 17. (Example 12) We have three filtering mechanism of which F_1 and F_2 specify the size of the filtered data explicitly whereas F_3 has a tuning parameter c, the choice of which controls the size of the filtered data. Figure 2.6.1 demonstrates how the choice of c affects the proportion of data we discard. This plot can be used to make an initial choice of c depending on the size of the available storage.

Regarding what we have already seen in Example 14 and Example 15 that even if we discard the same quantity of data, the quality of estimate depends on which observations we discard. Thus there should be an issue of optimal filtering mechanism which, besides controlling the size the of the filtered data, will search for the "best" possible option to throw away observations. The optimality of a filtering mechanism depends on two aspects: size of the filtered data and the efficiency of the parameter estimates. Generally this should turn out to be a trade-off: we need to strike balance between minimizing storage and maximizing efficiency. We shall see in chapter 5 how this trade-off can be tackled in a particular case while constructing optimal filtering mechanism for Markov chains. Further finding such optimal way to discard observations requires knowledge of the underlying population. For example, in Example 14, while finding the optimal value of c, we should take into consideration the expected proportion of missing observation as well as the standard error of the EM estimates. Calculation of both these quantities require that we know the distribution of the observed data. Similarly in Example 15, it is not possible to decide ideal groups for the data without some knowledge of the complete probability density.

Another important aspect in the construction of filter mechanism is to note that with respect to the estimation step of the proposed methodology the filtering mechanism is non-adaptive but the choice of the filtering mechanism itself should be adaptive to the underlying stochastic process which in turn is unknown, at least before the estimation step. In order to understand how the filtering mechanism should adapt to the underlying stochastic process, we can refer back to two previous examples:

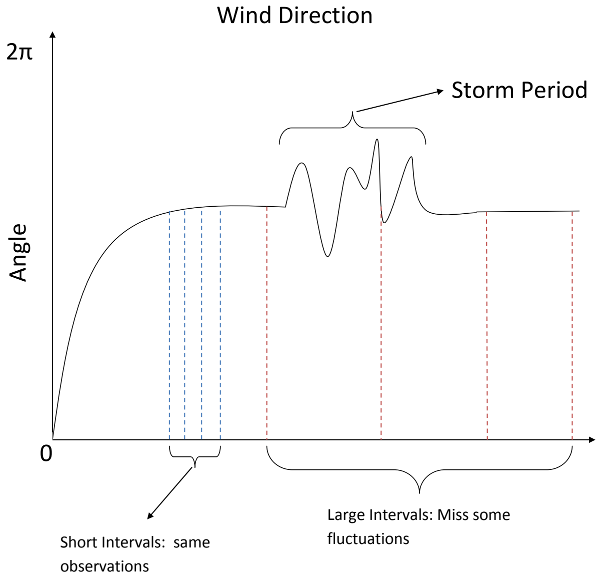
Example 18. (Example 6 continued) For most of the time in a day when no one enters the room, there will be no significant change in the pixel intensities and it only changes after some movement in the room.

If our filtering mechanism records observations at a high rate, there will be lots of information most of which are constant. On the other hand if the filtering mechanism stores observations at large intervals, we may miss certain variation in pixel intensities caused when someone enters the room. So our filtering mechanism should adaptively change its recording rate when there is a significant change in the underlying stochastic process (or population).

Example 19. (Example 7 continued) The angle of the wind directions mostly remains constant and fluctuates only during a storm or a windy patch. If our filtering mechanism records observations at a high resolution, we shall end up recording long stretches of constant observations. On the other hand if we record at a low resolution, then we may fail to detect the presence of any potential storm. The figure 2.6.2 illustrates this problem using a hypothetical data set of angles of wind directions. The filtering mechanism should adapt its recording rate according to the underlying population.

Thus the two parts of the proposed methodology: inference of parameters and the construction of filtering mechanism are somewhat entangled and circular. The design of an optimal filtering mechanism requires knowledge of the underlying population and estimating the parameters in the population requires a fully determined and fixed filtering mechanism. In order to find a solution, we take a look at the pilot survey example in Chapter 1.

Figure 2.6.2. Filter should adapt to change in underlying process



Plot of hypothetical data: angles change rarely in a day. Thus frequent observations lead to constant values whereas very less frequent observations may miss some significant change in the stochastic process.

Example 20. (Pilot Survey Example Continued) Suppose we have an infinite population from which it is possible to draw a random sample of large size. Further assume that we have the knowledge that our population assumes a parametric form $N(\mu, \sigma^2 = 0.01)$ and our objective is to estimate μ. Then if we draw a random sample of size n, then we can estimate μ by the sample mean $\bar{X}$ which has standard error $\frac{\sigma}{\sqrt{n}}$. Now suppose it is enough to keep the standard error less than

0.01. Then even if we have resources for collecting a large sample, we can restrict ourselves to a relatively smaller sample size n which will be good enough to produce estimates with satisfactory precision (so that $\frac{\sigma}{\sqrt{n}}$ is well below the tolerance limit). This situation can be viewed as if we are originally provided with a hypothetical sample of a very large size but we observe only a small part of the sample from the beginning, say $X_1, X_2, ..., X_n$, and discard the remaining samples $X_{n+1}, X_{n+2}, ...$. In practice, generally we do not perform this data trimming because for computing the statistic in this case, all we need to find is $\sum X_i$ which does not require much computational payload. However, there is another version of this problem in statistics where the process variance σ^2 is not known. In that case we perform a pilot survey or double sampling where we initially collect a portion of the available data and form a suitable estimate $\hat{\sigma}^2$ of the process variance and based on this estimate we determine the required sample size n so that $\frac{\hat{\sigma}}{\sqrt{n}}$ is below the tolerance limit. If this estimated sample size is $\hat{n}$, then even if we have scope of collecting more samples (say $\hat{n} + k$) we can satisfy ourselves with collecting only $X_1, X_2, ..., X_{\hat{n}}$ and effectively dropping $X_{\hat{n}+1}, X_{\hat{n}+2}, ..., X_{\hat{n}+k}$.

From this example, we can now form a protocol for filtering mechanism as :

(1) Collect initial sample of relatively small size and observe it completely.

(2) Based on this pilot sample, construct an estimate of the underlying probability distribution.

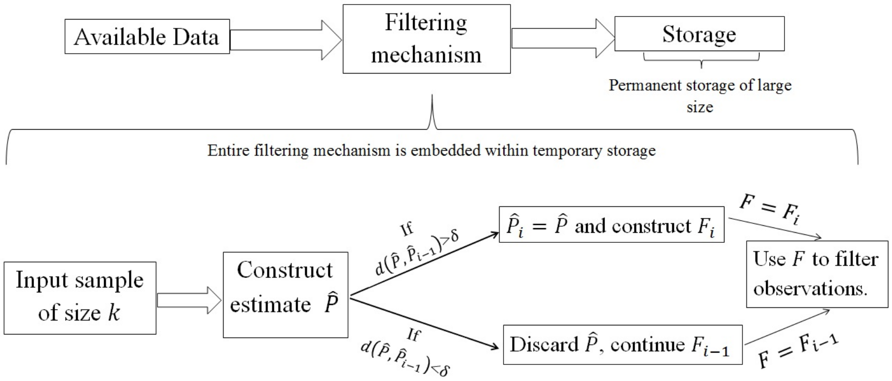
(3) Use the estimate to construct an optimal filtering mechanism.

(4) Apply the filtering mechanism to discard part of the data and store the rest permanently in the storage.

Coupled with this is the issue of adaptivity of the filtering mechanism. Moreover in most contemporary applications, data stream in as a continuous sequence of observations. Hence in order to be applicable in practice, our filtering mechanism should be online, continuously adapting itself with the data and to be embedded as a preprocessing step in the data storage pipeline. Keeping all these things in mind, we shall propose a sequential procedure which fits well in this online filtration paradigm and is also adaptive in nature.

2.6.1. The adaptive data filtering mechanism: Suppose we have two kinds of storage: a permanent storage with relatively large size and a temporary storage of smaller size. When the stream of observations starts, we shall store the data completely up to a number of observations (say k) in the temporary memory and based on those observations we derive an estimate $\hat{P}_1$ of the underlying probability distribution. Based on this estimate we can construct the optimal filtering mechanism F_1 which can be used to filter the observations. Based on each such chunk of k observations, we keep on deriving a temporary estimate $\hat{P}$ of the probability distribution. If the estimate $\hat{P}$ does not change significantly, we continue with the same filtering mechanism, otherwise we shall consider $\hat{P}$ as our new estimate to construct a new filtering mechanism F. We can detect the change in the probability distribution using some suitable distance measure $d : \Theta \times \Theta \to [0, \infty)$. In

Figure 2.6.3. The data storage pipeline


Available Data
Filtering mechanism
Storage
Permanent storage of large size
Entire filtering mechanism is embedded within temporary storage
Input sample of size k
Construct estimate $\hat{P}$
If $d(\hat{P},\hat{P}_{i-1})>\delta$
$\hat{P}_i = \hat{P}$ and construct F_i
If $d(\hat{P},\hat{P}_{i-1})<\delta$
Discard $\hat{P}$, continue F_{i-1}
$F = F_i$
$F = F_{i-1}$
Use F to filter observations.

The filtering mechanism is constructed using a crude estimate $\hat{P}$ based on a pilot sample of very small size. Moreover the filtering mechanism is adaptive to significant change in the underlying stochastic process.

particular, if we are considering a parametric family $P = P_\theta$, then one can choose $d = ||\theta - \hat{\theta}||_p$ for some p. If this distance $d(P, \hat{P})$ crosses a threshold δ, we consider a change in the underlying stochastic process and the new estimate $\hat{P}$ is used to update the filtering mechanism F.

Figure 2.6.3 shows the data storage pipeline we just discussed. This way, our filtering process remains adaptive to the incoming data and subsequently providing a considerable reduction of data to be stored in the permanent memory. In practice, this entire filtering mechanism can be embedded in a temporary storage and programmed within a micro-controller which can be fit directly to a sensor mechanism collecting the data.

CHAPTER 3

Data reduction under independence

3.1. Introduction

In recent years there has been a huge explosion in the variety of sensors as well as in the dimensionality of the data produced by these sensors. This is true for a large number of applications ranging from imaging to other scientific areas. Such types of data are usually of large dimensions or we may require to observe a large number of samples. The total amount of data produced by the sensors is much more than the available storage. So we want to store a subset of the information and reconstruct as much as possible of the entire information from it. Instead of working with the entire data we observe certain linear combinations of the observations and try to estimate the parameter from them. In general the problem is ill posed in the sense that unique estimates of the parameter may not exist. But we shall assume sparsity of the parameter to get a unique solution. Sparsity is a common phenomenon in nature and hence is a reasonable assumption to make in many real applications. For example, while working with astronomical data, high resolution telescope images are generally sparse as only few pixels correspond to the stars, the rest being dark.

We shall work with independent normal samples with different means. Thus the problem can be viewed as a single sample multivariate normal problem or independent samples from different univariate normal. The methods suggested here will be a lossy compression algorithm in the sense we will lose some information about the parameter in the process but the loss in information is compensated by the advantage due to data reduction not sacrificing reasonable inference of the parameter.

Our main weapon will be the EM algorithm [**18**] which is a powerful tool for maximum likelihood estimation in incomplete data problems. Even in situations where there are no actual missing data, the given problem can be reformulated as a missing data problem and can be tackled with the EM algorithm. Unlike the other uses of EM algorithm where missingness occurs naturally as a problem, here we deliberately incorporate missingness to reduce the number of observed samples. The EM algorithm is computationally straightforward when there is no restriction on the parameters. Sparsity of the parameter, however, imposes certain restrictions on the parameter for which solutions may get complicated. Kim and Taylor [**30**] worked with parameter under linear restrictions and proposed a restricted EM algorithm using constrained maximization. In the present situation sparsity imposes linear restrictions on the parameter which are not known in advance. Thus the methods of Kim and Taylor cannot be applied directly to produce estimates. In this paper we propose new methods based on the restricted EM algorithm of Kim and Taylor to apply in this sparse data situation. The modifications, however, do not affect the desirable theoretical properties of the algorithm.

Similar data compression techniques using linear combination of the observations already exist in signal sampling paradigm, for example, Compressive Sampling that uses a different data recovery algorithm. Signal sampling methods based on the Nyquist-Shannon sampling theorem [**43**] often result in too many samples. In contrast compressive sampling try to approximate the true signal with few linear combinations. We compare our method with the compressive sampling technique using the same data acquisition protocol but estimating the parameter with our proposed approach.

Section 2 gives a brief literature review of works relevant to this paper. Section 3 describes the setup of the problem. Section 4 deals with the identifiability issues of the parameter that arise due to dimension reduction. Section 5 briefly describes the conventional method of data recovery in compressive sampling. Section 6 introduces our new approaches "All Subspace Restricted EM Algorithm (ASREM)" and "Estimated Subspace Restricted EM Algorithm (ESREM)". Section 7 discusses some theoretical properties of the new methods. Section 8 applies the new algorithms to hypothesis testing and finding confidence interval. Section 9 compares the new approaches to the conventional compressive sampling using simulation. Section 10 illustrates the application of ESREM in a real life dataset. Section 11 gives some conclusions and possible extensions regarding the problem. Section 12 is the appendix which contains the technical details and proofs of the theorems stated in this paper.

3.2. Literature Review

Dempster, Laird and Rubin [**18**] introduced EM algorithm for maximum likelihood estimation in incomplete data problems. Restricted parameter problem is very common in many statistical applications. Kim and Taylor [**30**] developed the EM algorithm for maximum likelihood estimation for incomplete data under linear restrictions on the parameter. Shi, Zheng and Guo [**44**] studied EM algorithm under inequality restrictions on the parameter. Tian, Wang and Tan [**46**] described restricted EM algorithm for multivariate normal models under regression setup. [**45**] applied EM type algorithms for restricted MLE in univariate normal distribution with known and unknown variance. In this paper we assume the parameter to be sparse and develop EM type algorithms to estimate the parameter and construct hypothesis tests and confidence intervals. Applying EM algorithm in hypothesis testing and confidence interval follows from the early works of Louis [**34**] and Meng and Rubin [**39**]. These approaches give symmetric Wald-type intervals based on asymptotic dispersion matrix.

An already available technique for similar situations is called compressive sampling. It was introduced in signal sampling paradigm by Donoho [**19**] and has been further studied by Candes [**11**]. [**6**] and [**13**] and [**40**] give a good description of the compressive sampling problem.

3.3. Setup

In this paper we assume that our data $x \in \mathbb{R}^n$ are coming from a $N_n(\mu, \sigma^2 I_n)$ population where $\mu = (\mu_1, \mu_2, ..., \mu_n)' \in \mathbb{R}^n$ is unknown and σ^2 is known. The complete data x are not stored. Instead we only store $m\,(\ll n)$ linear combinations of x, which we call y. Then we can write $y = \phi x$ where ϕ is a fixed $m \times n$ matrix, whose rows give the coefficients of the m linear combinations. We choose ϕ in such a way that $rank[\phi] = m$. Also the observation process is non-adaptive in the sense that ϕ does not depend in any way on the complete data x. Further μ is assumed to be sparse so that at most m elements of μ are nonzero. We shall treat x as missing, and we shall try to estimate μ by the EM algorithm based on the observed data, y. In general we do not know *apriori* which of the μ_i's are nonzero. This will make maximization of the likelihood a little difficult. Moreover since $m \ll n$, the problem does not have a unique solution in general. But, fortunately, sparsity of the parameter will come to our rescue.

3.4. Identifiability

The complete data $x \in \mathbb{R}^n$ which is unobserved has mean $\mu = (\mu_1, \mu_2, ..., \mu_n)'$. Since x has $N_n(\mu, \sigma^2 I_n)$ distribution all the μ_i's would have been estimable if we could have stored x. Instead we observe $y \in \mathbb{R}^m$ which has distribution $N_m(\phi\mu, \sigma^2\phi\phi')$ where $rank[\phi] = m$. From the observed data we can estimate the m components of $\phi\mu$, that is, m linear equations involving μ_i's. Hence we need additional $n - m$ linear restrictions on μ so that all μ_i's are estimable. These additional restrictions will be provided by the assumption of sparsity of the parameter. The observed log-likelihood is

$$\ell_{obs}(\mu) = constant - \frac{1}{2}(y - \phi\mu)^{'}(\sigma^2\phi\phi^{'})^{-1}(y - \phi\mu).$$

Setting $\frac{\partial}{\partial\mu}\ell_{obs}(\mu) = 0$ we get

$$(\phi^{'}V^{-1}\phi)\mu = \phi^{'}V^{-1}y \tag{3.4.1}$$

where $V = \phi\phi^{'}$. Since

$$rank[(\phi^{'}V^{-1}\phi)_{n\times n}] = m \ll n$$

this system of equations does not have a unique solution and we need to have a linear restriction of the form $H\mu = 0$ such that

$$rank[\begin{bmatrix}(\phi^{'}V^{-1}\phi)_{n\times n}\\ H\end{bmatrix}] = n.$$

Such a linear restriction matrix H such that $rank[H] = n - m$ and $\mathscr{R}(H)\bigcap\mathscr{R}(\phi^{'}V^{-1}\phi) = \{0\}$ (where $\mathscr{R}$ denotes the rowspace of a matrix) will be decided from the sparsity assumption that at least $n - m$ components of μ are zero. However, since we do not know the matrix H explicitly, the observed likelihood cannot be maximized directly. This task will be accomplished by the EM- based algorithms, which we shall discuss in Section 3.6.

3.5. An Existing Approach

As far as we know our approach of using EM algorithm for data compression has not been proposed earlier. However, for the sake of comparison we describe here an existing approach called Compressive Sampling (CS) which is somewhat similar, though not related with our EM approach. Compressive Sampling deals with a general problem of reconstructing a vector $x \in \mathbb{R}^n$ from $m(\ll n)$ linear measurements $y = Ax$. If the original signal $\mathbf{x}$ is sparse then it can be recovered exactly by solving

$$\min_{x^* \in \mathbb{R}^n} \parallel x^* \parallel_{\ell_1} \text{ subject to } Ax^* = y.$$

This technique is also known as basis pursuit. However in real life applications observations are subject to measurement error or noise. So instead we shall mainly focus on what is called Robust Compressive Sampling [**11, 13**] which deals with signals associated with noise. Here it is assumed that the true signal x is sparse and the data collected by a measurement system consists of some linear combinations of the signals

$$y = Ax + e$$

where A is a measurement matrix (also called sensing matrix) which is chosen beforehand. e is the error which is assumed to be bounded or bounded with high probability. Compressive Sampling(CS) designs a reconstruction algorithm to recover the original data x from the measurements y. Here we shall describe reconstruction using $\min -\ell_1$ with quadratic constraints, but one can also consider other reconstruction algorithms as well [**13**]. We note that the recovery algorithm addresses the problem of solving for x when the number of unknowns (i.e. n) is much larger than the number of observations (i.e. m). In general this is an ill-posed problem but CS theory provides certain conditions on A which allows accurate estimation. One such popularly used property is the Restricted Isometry Property (RIP).

Definition 21. The matrix A is said to satisfy the restricted isometry property of order k with parameter $\delta_k \in [0, 1)$ if

$$(1-\delta_k) \parallel \theta \parallel_2^2 \leq \parallel A\theta \parallel_2^2 \leq (1+\delta_k) \parallel \theta \parallel_2^2$$

holds simultaneously for all sparse vectors θ having at most k nonzero entries. Matrices with this property are denoted by RIP(K, δ_k).

The following theorem shows that matrices satisfying RIP will yield accurate estimates of x with the help of recovery algorithms [**49**].

Theorem 22. *Let A be a matrix satisfying RIP$(2k, \delta_{2k})$ with $\delta_{2k} < \sqrt{2}-1$ and let $y = Ax + e$ be a vector of noisy observations , where $\parallel e \parallel_2 \leq \epsilon$. Let x_k be the best k-sparse approximation of x , that is , x_k is the approximation obtained by keeping the k largest entries of x and setting others to zero. Then the estimate*

$$\hat{x} = \arg\min_{x \in \mathbb{R}^n} \parallel x \parallel_1 \ \textit{subject to} \ \parallel y - Ax \parallel_2 \leq \epsilon \tag{3.5.1}$$

obeys

$$\| x - \hat{x} \|_2 \leq C_{1,k}\epsilon + C_{2,k}\frac{\| x - x_k \|_1}{\sqrt{k}} \tag{3.5.2}$$

where $C_{1,k}$ and $C_{2,k}$ are constants depending on k but not on n or m.

The reconstruction in (3.5.1) is equivalent to

$$\hat{x} = \arg\min_{x \in \mathbb{R}^n} \frac{1}{2} \| y - Ax \|_2^2 + \nu \| x \|_1 \tag{3.5.3}$$

where $\nu > 0$ is a regularization parameter which depends on ϵ.

There are many ways of constructing RIP matrices from $m \times n$ random matrices [**7**]. Consider an $m \times n$ random matrix $A = (a_{ij})$ and the following choices of a_{ij}:

- $a_{ij} \sim N(0, \frac{1}{m})$
- $a_{ij} = \begin{cases} +1/\sqrt{m} & \text{with probability } \frac{1}{2} \\ -1/\sqrt{m} & \text{with probability } \frac{1}{2} \end{cases}$
- $a_{ij} = \begin{cases} +3/\sqrt{m} & \text{with probability } \frac{1}{6} \\ 0 & \text{with probability } \frac{2}{3} \\ -3/\sqrt{m} & \text{with probability } \frac{1}{6} \end{cases}$

Then it can be shown [**7**] that A satisfies RIP(K, δ_K) with high probability for any integer $K = O(m/\log n)$. In our work we shall use $A = (a_{ij})$ where $a_{ij} \sim N(0, \frac{1}{m})$ to generate the observed data in the simulation. However, we shall use the symbol ϕ in place of A in the following sections.

3.6. Our Approach

As mentioned already in section 3, we shall treat x as the complete data and y as the observed data. As a natural tool of missing data analysis, we shall apply EM algorithm for the estimation of the parameter. Each iteration of EM algorithm consists of an E-step which requires computation of the expected complete data log-likelihood (with respect to the conditional distribution of the complete data given the observed data) and an M-step where we maximize the same with respect to the parameter. Following our setup, the complete data log-likelihood is given by

$$\begin{aligned}\ell(\mu) &= k_1 - k_2(x-\mu)^{'}(x-\mu) \\ &= k_1 - k_2\sum_{i=1}^{n}(x_i-\mu_i)^2\end{aligned}$$

where k_1 and k_2 are known constants. Then after t iterations in the EM algorithm, at the E-step we compute

$$Q(\mu) = E(\ell(\mu)|y, \mu^{(t)})$$

where the expectation is computed with respect to the conditional distribution of $x|y, \mu^{(t)}$ and $\mu^{(t)}$ is the value of the parameter after t iterations.

At the M-step, we try to maximize $Q(\mu)$ with respect to μ. Here sparsity of the parameter imposes certain restrictions on the parameter space, which is a *strict* subset S of $\mathbb{R}^n$ where

$$S = \{\mu : \text{at most } m \text{ elements of } \mu \text{ are nonzero}\}.$$

This subset can be decomposed as $S = \bigcup_{i=1}^{\binom{n}{m}} S_i$, where

$$S_i = \{\mu : \text{at most } m \text{ specific elements of } \mu \text{ are nonzero}\}.$$

We note that each S_i is a linear subspace of $\mathbb{R}^n$.

The difficulty of the problem is that we do not know in advance in which of these subspaces μ lies. We shall describe here two approaches, which we call "**All Subspace Restricted EM Algorithm (ASREM)**" and "**Estimated Subspace Restricted EM Algorithm (ESREM)**" to circumvent the problem. Both these approaches make use of a variant of the EM algorithm called the Restricted EM Algorithm (henceforth we shall refer to it as REM) of Kim and Taylor [**30**]. We first present the theory behind our proposed algorithms. Implementational issues will be discussed later.

3.6.1. All Subspace Restricted EM Algorithm (ASREM).

In ASREM, we maximize $Q(\mu)$ over each of those subspaces at each M-step. Then after t iterations, the M-step proceeds as follows to compute $\hat{\mu}^{(t+1)} = \arg\max_{\mu\in S} Q(\mu)$.

We first compute

$$\hat{\mu}^{(t+1)}(S_i) \equiv \left(\hat{\mu}_1^{(t+1)}(S_i), \hat{\mu}_2^{(t+1)}(S_i), ..., \hat{\mu}_n^{(t+1)}(S_i)\right)' = \arg\max_{\mu\in S_i} Q(\mu).$$

for each i.

We then define $\hat{\mu}^{(t+1)} = \hat{\mu}^{(t+1)}(S_i)$ such that

$$Q(\hat{\mu}^{(t+1)}(S_i)) \geq Q(\hat{\mu}^{(t+1)}(S_j)) \; \forall j = 1, 2, \cdots, \binom{n}{m}.$$

Proceeding in this way, we iterate until convergence is attained to the desired level of accuracy.

3.6.2. Estimated Subspace Restricted EM Algorithm (ESREM). In ESREM, instead of maximizing the $Q(\mu)$ over all possible subspaces as described in the previous subsection, we try to estimate the subspace where μ lies. This estimation is done before the steps of the EM algorithm. After estimating the subspace, we maximize $Q(\mu)$ at each M-step on that estimated subspace.

Let S_μ be the subspace where μ lies , that is

$$S_\mu = \big\{(x_1, x_2, \dots, x_n) \in \mathbb{R}^n : \forall i \ (\mu_i = 0 \ \Rightarrow x_i = 0)\big\}.$$

We hope that if we find the unrestricted maximizer of $Q(\mu)$ in each M-step of the EM algorithm (henceforth called the unrestricted EM), that is, if we find

$$\hat{\mu}^{un} = \arg\max_{\mu \in \mathbb{R}} Q(\mu),$$

then the unrestricted EM estimate $\hat{\mu}^{un}$ should lie close to S_μ. Hence we find which components of $\hat{\mu}^{un}$ are significant, and take the other (insignificant) components to be zero. We take the corresponding subspace as estimate of S_μ. To test which components of $\hat{\mu}^{un}$ are significantly different from zero we write $V = \phi\phi^{'}$ and $P = (\phi^{'}V^{-1}\phi)^{+}\phi^{'}V^{-1}$, we choose the test statistics to be $\tau_i = |\frac{\hat{\mu}_i^{un}}{\sigma\sqrt{s_{ii}}}| \ \forall i = 1, 2, \cdots, n$, where $s_{ii} = i^{th}$diagonal element of $PVP^{'}$.

Then we estimate S_μ as

$$\hat{S}_\mu = \big\{(x_1, x_2, \dots, x_n) \in \mathbb{R}^n : \forall i \ (\tau_i \leq z_{\alpha/2} \ \Rightarrow x_i = 0)\big\}. \tag{3.6.1}$$

But this may sometimes contradict our initial assumption that at most m components of μ are nonzero. If the cardinality of the set $\{\, i \ : \tau_i > z_{\alpha/2}\}$ is more than m, then we order the τ_i's in increasing order of magnitude, say $\tau_{(i)}, i = 1, 2, \cdots n$ and take the components corresponding to the $\tau_{(1)}, \tau_{(2)}, \cdots, \tau_{(m)}$ to be significant. In that case the estimated subspace looks like

$$\hat{S}_\mu = \big\{(x_1, x_2, \ldots, x_n) \in \mathbb{R}^n : \, \forall i \; \tau_i \notin \{\tau_{(1)}, \tau_{(2)}, \cdots, \tau_{(m)}\} \, \Rightarrow x_i = 0\big\}.$$

It may be noted that $\hat{S}_\mu$ is one among the $\binom{n}{m}$ subspaces S_i's. With this new estimated subspace we apply our original restricted EM algorithm as in the previous subsection as follows.

After t iterations in EM algorithm we have, the M-step of the EM algorithm as

- **M-step**: We find

  $$\arg\max_{\mu \in \hat{S}_\mu} Q(\mu)$$

  and take the maximizer as the new estimate of μ after the $(t+1)^{th}$ iteration, that is , $\hat{\mu}^{(t+1)}$.

The iterations are continued until convergence is attained.

3.7. Some Discussions

This section deals with some properties of the proposed algorithms and some important points regarding their application to the estimation of the parameter.

3.7.1. Theoretical properties.

In this subsection we shall prove that both ASREM and ESREM share the property of convergence of the observed likelihood. We shall also work on the measure of closeness of the estimated and the original parameter values in case of ESREM. These results show that the algorithms suggested in this paper will produce a reasonable estimate of the parameter.

3.7.1.1. *Nondecreasing nature of the observed likelihood with each iteration.* Kim and Taylor [**30**] showed that REM being adaptation of the EM and GEM algorithms share some of the properties of EM and GEM algorithms. In their paper they showed that the observed log-likelihood, denoted by $\ell_{obs}(\mu)$, is nondecreasing with each iteration of the restricted EM algorithm as the Q function satisfies $Q(\hat{\mu}^{(t+1)}) \geq Q(\hat{\mu}^{(t)})$ at the $(t+1)^{th}$ stage. In our ASREM and ESREM we apply this REM with slight modifications. Thus the nondecreasing property of the observed likelihood should also be retained in our algorithms. The following theorem states that:

THEOREM 23. *Both in ASREM and ESREM the observed log-likelihood* $\ell_{obs}(\mu)$ *is nondecreasing with each iteration, that is,* $\ell_{obs}(\hat{\mu}^{(t+1)}) \geq \ell_{obs}(\hat{\mu}^{(t)})$.

3.7.1.2. *Measure of closeness between the estimated and original values in ESREM.* We take $\| \hat{\mu} - \mu \|_{l_2}$as a measure of closeness between the original and the estimated values of the parameter. The expected value of the discrepancy, $E[\| \hat{\mu} - \mu \|_{l_2}]$, will give an idea about the goodness of the estimate. The following theorem, in this connection, finds an upper bound to this expected discrepancy provided the subspace is correctly chosen.

Theorem 24. *If $\mu \in \hat{S}_\mu$ then*

$$E[\| \hat{\mu} - \mu \|_{l_2}] \leq \sigma \sqrt{\sum_{i=1}^{n} s_{ii}}$$

where $\sigma^2 s_{ii} = Var(\hat{\mu}_i^{un})$.

$\sigma^2 \sum_{i=1}^{n} s_{ii}$ is the trace of the dispersion matrix $\sigma^2 PVP'$ and is a measure of the total variation in $\hat{\mu}^{un}$. Thus the total variation in $\hat{\mu}^{un}$ provides an upper bound for the measure of closeness of the estimated and true value of the parameter provided the subspace is correctly estimated.

3.7.2. Implementational Issues. Both the algorithms ASREM and ESREM suggested above have their own implementational complexities and may not be applicable in certain situations.

- The ASREM requires the maximization of $Q(\mu)$ over $\binom{n}{m}$ subspaces and then choose the one for which it is maximum at the M-step of each EM iteration. This is computationally expensive and practically impossible to implement for large n. Hence for large values of n we suggest to apply ESREM which identifies a particular subspace where μ is most likely to belong , and then finds the maximum over that subspace in each M-step.
- In ESREM the tests of significance of the components of $\hat{\mu}^{un}$ give us an idea regarding the subspace where the true μ lies. But in certain situations these tests may favor the alternate hypotheses and hence can identify the wrong subspace. The

> tests find the components of $\hat{\mu}^{un}$ that are insignificant compared to the others. If the true μ lies very close to the origin, that is, if the non zero components of μ are close to zero (that is the signal to noise ratio is low), then all the components of $\hat{\mu}^{un}$ will also be close to zero. This will inflate the value of the test statistics τ_i and tend to reject the null hypotheses. The sparse nature of μ will not be captured and the true subspace will not be identified.

In theorem 3 we showed that $\sigma\sqrt{\sum_{i=1}^{n} s_{ii}}$ is an upper bound to $E[\| \hat{\mu} - \mu \|_{l_2}]$ provided the estimated subspace is the true subspace where μ lies. Thus given the distribution of the unrestricted EM estimate $\hat{\mu}^{un}$ it is possible to get an idea of the error in estimation.

3.8. General Linear Hypothesis

In this section we will apply ASREM and ESREM to construct tests and confidence intervals for the parameters of interest. The methods developed here are important because EM algorithm does not automatically produce the dispersion matrix for the parameters and additional steps are needed to construct it (Louis 1982; Meng and Rubin 1991). Kim and Taylor showed in their paper how REM can be applied to form test statistics and confidence intervals. The same approach can be adapted in the case of ASREM and ESREM also.

Suppose we want to test the hypothesis:

$$H_0 : \ L\mu = \beta$$

where L is a $n \times n$ matrix and β is a $n \times 1$ vector. We note that the null hypothesis imposes certain additional linear restrictions on the parameter. Hence both ASREM and ESREM can be modified to find the estimates under these additional restrictions. Let $\hat{\mu}_{H_0}$be the estimate obtained from ASREM or ESREM under the additional linear restrictions $L\mu = \beta$. Then the likelihood ratio test statistic is

$$r = -2[\ell_{obs}(\hat{\mu}_{H_0}) - \ell_{obs}(\hat{\mu})].$$

Under suitable regularity conditions [**15**], r has asymptotically χ^2 distribution with degrees of freedom decided by the difference in the number of independent parameters in the full model and the model under the null hypothesis.

The confidence interval for any μ_i can be constructed using the likelihood ratio method. Suppose we consider a special case of the general linear hypothesis as $H_{0i} : \mu_i = \mu_0$. Then the likelihood ratio test criterion r rejects the null hypothesis at level α if $r > \chi^2_{\alpha,1}$ and as such a $100(1-\alpha)\%$ confidence interval for μ_i can be defined as

$$\left\{\mu_0 : \ r > \chi^2_{\alpha,1}\right\}.$$

3.9. Simulation Study

This section compares the algorithms developed in this paper with compressive sampling with the help of simulated data. But first we verify the convergence of $\hat{\mu}^{un}$ to the sparsest solution as claimed before. The performance of the new proposed approaches will be studied using simulation technique where we will investigate to what extent we can

TABLE 3.9.1. Simulation study to check the minimum norm solution

Initial estimate $\hat{\mu}^{(1)}$	L_1 norm of $\hat{\mu}^{un}$
(0.0001,0.0001,0.0001,0.0001)	10.5667
(12.52,22.76,35.98,67.72)	38.9358
(10.5,11.25,25.62,19.74)	27.8503

Simulation study showing that the minimum norm solution is attained when the initial estimate of μ is closest to 0.

reduce the dimension of the observed data using ESREM in order to have a fair reconstruction of the parameter.

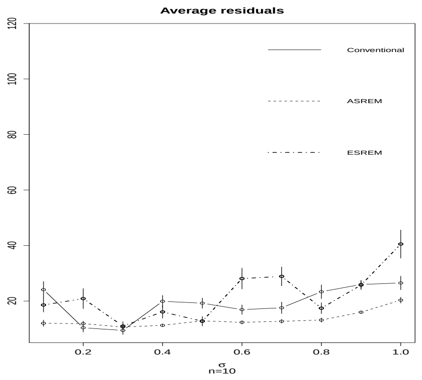
3.9.1. Convergence of the Unrestricted EM estimate in ESREM. Here we empirically verify that in the unrestricted EM algorithm the EM estimate of μ converges to the sparsest solution of equation (3.12.2) if we take our initial estimate as 0 (or very close to 0). We take different initial estimates of μ randomly and check the L_1 norm of the final estimates $\hat{\mu}^{un}$ in each case. For demonstration we work with $n = 4$ and the results are shown in table 3.9.1. We find that we reach the minimum norm solution if the initial estimate of μ is taken close to 0.

3.9.2. Comparison with Compressive Sampling. Here we compare the accuracy of ASREM and ESREM with that of compressive sampling. We compute $\parallel x - \widehat{x} \parallel_{l_2}$, the measure of closeness between the original and the reconstructed signal. We note that there is a difference in the setup of the data in our methods as compared to the conventional compressive sampling. The conventional approach reconstructs the signal (data) x whereas in our approaches we reconstruct (estimate) what is called the true signal (free from noise) μ. Hence for comparison with the conventional compressive sampling approach we reconstruct signals repeatedly from same population using the conventional approach and average out the residuals to remove the effect of the noise.

For the comparison of approaches we adopt the following steps:

- We set the actual number of observations n and the observed number of observations m. k, the maximum number of nonzero components in μ, is taken to be equal to m (maximum possible value), that is, we do not use any prior information about the number of nonzero components in μ.
- We fix a μ such that its first 4 components are 5 and the rest are zero.
- We start with a value of σ between 0.1 and 1 (Recall our assumption that σ is known).
- **Assessing the Conventional Approach:** We generate data x from $N_n(\mu, \sigma^2 I_n)$ and reconstruct $\widehat{x}$ using (3.5.3) from the conventional approach and find $\parallel x - \widehat{x} \parallel_{l_2}$. This process is repeated 1000 times to find the residuals in each case and then we compute the mean residual $\frac{1}{1000}\sum\limits_{i=1}^{1000} \parallel x_i - \widehat{x}_i \parallel_{l_2}$ to remove the effect of randomness and get a measure of closeness among the original and reconstructed μ.
- **Assessing Our Approaches (ASREM and ESREM):** We again generate data x from $N_n(\mu, \sigma^2 I_n)$. We apply the ASREM (wherever possible) and the ESREM to reconstruct μ and find

FIGURE 3.9.1. Average residuals for three algorithms



The average residuals in case of all the three algorithms are shown for different values of σ with $n = 10$ which shows ASREM works better than other methods.

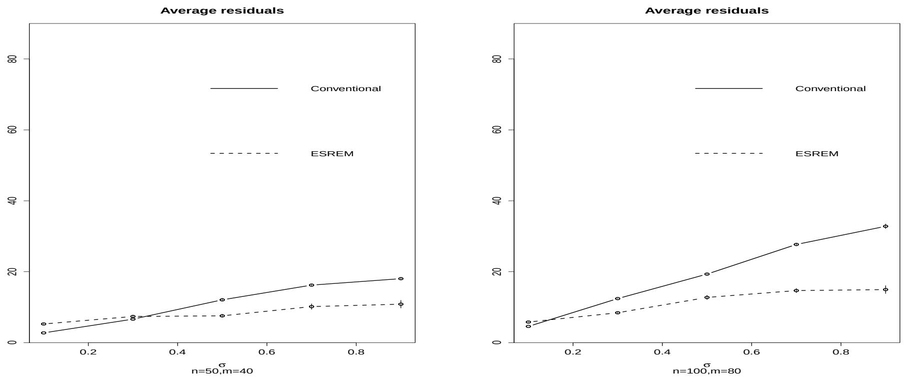
$\| \mu - \hat{\mu} \|_{l_2}$ as a measure of closeness between the original and estimated values.

- For each value of σ in we repeat the process of assessing the conventional approach as well as our approaches 10 times each to get the average residual and standard error of the residuals for each of the conventional and the proposed algorithms.
- We repeat the above procedures for different values of σ in $[0.1, 1.0]$ and plot the mean residuals along with the standard error bars.

For small values of n we study the average residuals for different values of σ for all the three algorithms .

For $n = 10$, the figure 3.9.1 shows that ASREM works uniformly best for different values of σ as compared to the conventional method and ESREM. Though for small values of n ASREM performs better as compared to the other methods, but this algorithm cannot be applied for large values of n. In such cases we turn our attention towards

FIGURE 3.9.2. Average residuals for two algorithms



The average residuals in case of the two algorithms are shown for different values of σ with $n = 50$ and 100 which shows ESREM works better than the conventional method.

ESREM. In figure 3.9.2 we plot the average residuals using ESREM and the conventional method for both $n = 50$ and $n = 100$ and find that ESREM works relatively better than the conventional method for different values of σ.

We also compared the computation time taken by ESREM with that of the conventional compressive sampling algorithm. With our limited computing facility, both the algorithms were set to run with the R programming environment in a standard machine. The computation of the inverse of the measurement matrix and other related matrix products are not included in the computation time because the measurement process being non-adaptive, those matrix computations can be done beforehand and stored much before the observation process starts. Taking $\sigma = 0.1$, we report in Table 3.9.2, the average system time (in seconds) taken by both the algorithms in case of $n = 50, m = 40$ and $n = 100, m = 80$.

TABLE 3.9.2. Comparison of System time

	$n = 50, m = 40$	$n = 100, m = 80$
ESREM	0.06920431	1.405794
Conventional Algorithm	3.371517	19.12753

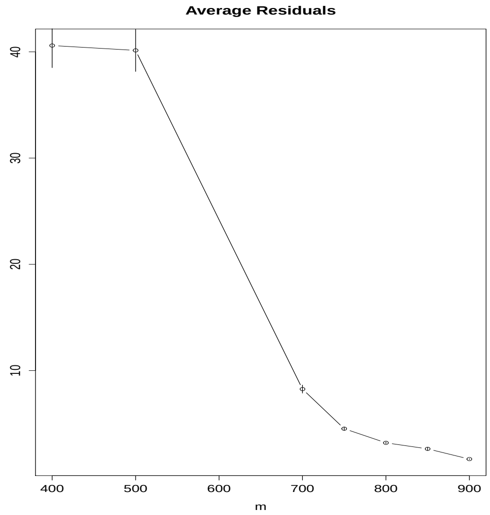
The values show that ESREM can be used as a quick algorithm for the estimation of the parameter. It performs comparably with the conventional CS algorithm for moderate error levels and outperforms it for larger error levels.

3.9.3. Performance of ESREM in data reduction. As mentioned earlier n is the size of the complete data and m is the number of the observed linear combinations. Thus the value of $\frac{m}{n}$ is an important point of consideration. It signifies the sampling fraction, that is, to what extent we can reduce the data to get good estimate. We fix $n = 1000$ and with $\sigma = 0.001$ we plot the average residuals with different values of m using ESREM in figure 3.9.3.

The plot shows that the average residuals while estimating μ are very small if we take m more than or equals to 750. Hence ESREM is found to work well even for m as small as 750, that is, at this variance level we can afford to store only 75% of the original sample size to get a good estimate. Obviously the performance of ESREM will decrease if we increase the value of σ.

Thus we find that both the ASREM and ESREM perform well in reconstructing the population parameter. Naturally, ASREM works best if it is not computationally prohibitive for a given problem. For

FIGURE 3.9.3. Average residuals in ESREM for different m



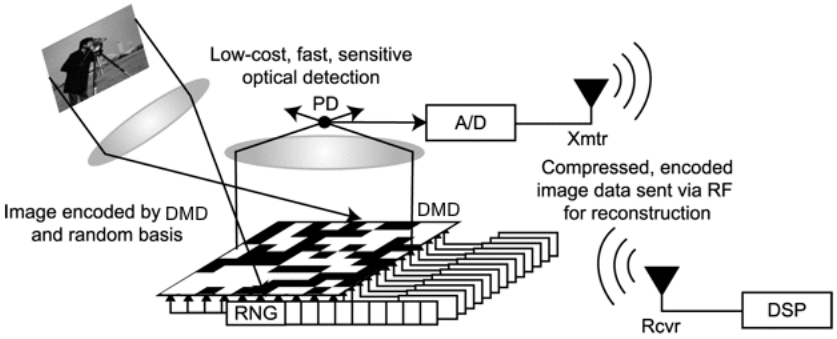
The plot shows the change in average residuals for different values of m in case of ESREM for $n = 1000$ and $\sigma = 0.001$.

moderate to large dimensional problems which are common in practice ESREM is suggested in order to get a reasonable estimate of the parameter. For both ASREM and ESREM the value of the average residuals will depend on σ and the sampling fraction, $\frac{m}{n}$.

3.10. Practical Example

One of the most well-known examples of a CS imager is the Rice single-pixel camera developed by [**21**]. Instead of first collecting the n pixel values, the single-pixel compressive digital camera directly observes m random linear measurements of a scene. Here the incident light field corresponding to a target image (say x) is reflected off a digital micromirror device (DMD) which consists of n mirrors whose movements are controlled by a random number generator. This reflected light is then collected by a second lens and then focused onto

Figure 3.10.1. Single pixel camera

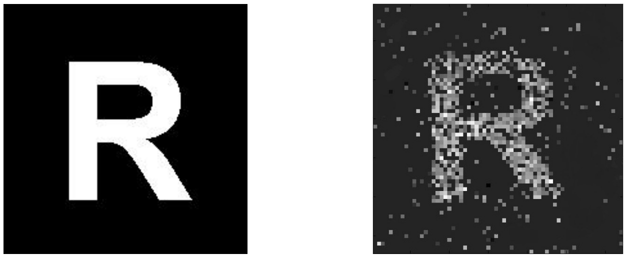

Low-cost, fast, sensitive optical detection
PD
A/D
Xmtr
Compressed, encoded image data sent via RF for reconstruction
Image encoded by DMD and random basis
DMD
RNG
Rcvr
DSP

a single photodiode (which is the single pixel). Each of the n mirrors of DMD can be independently oriented either towards the photodiode (which corresponds 1) or away from it (which corresponds 0).

The n mirror orientations are set in a pseudo random 0 or 1 pattern by the random number generator to collect the data. This sets the i^{th} row of the measurement matrix ϕ. The voltage at the photodiode is then observed which is the i^{th} observed linear combination y_i. This process is repeated m times to obtain all the observed linear combinations $y = (y_1, y_2, \cdots, y_m)'$. The measurement matrix ϕ thus formed has each row containing 0 or 1 in a random manner.

The data used in this paper are obtained from "Rice Single-Pixel Camera Project, http://dsp.rice.edu/cscamera". The figures in 3.10.2 illustrate a target object (on the left) which is a image of the letter "R" and a reconstructed image (on the right) of the same using 40% random measurements than the reconstructed pixels. The reconstructed image is a 64×64 black and white image of the letter "R" ($n = 4096$) reconstructed from $m = 1638$ random measurements taken by the camera.

FIGURE 3.10.2. Original and reconstructed images



The figure on the left is the original image of the letter "R" and the figure on the right is the reconstruction from 1638 random measurements.

The reconstruction is performed by ESREM. In the theory behind ESREM the error variance σ^2 was assumed known. However in this example the error variance is not known. Instead we are given the error bound ϵ (in 3.5.1) which is 0.01. Since for our approach we assume that the error is normally distributed with zero mean, so$| e |< 3\sigma$ with high probability and hence we approximate the value of σ as 0.00333.

3.11. Conclusions

The present paper employs EM algorithm for data compression in a iid setup. Unlike the classical uses of EM algorithm where missingness appears naturally, here missingness is introduced deliberately to reduce the number of observed samples. While the estimation methods developed here seem reasonably good, one should note that these methods are only for those situations where the original sample is too large to store or observe. The entire work is based on the assumption that the EM estimate should be close to the true value of the parameter. However, this may not be always correct due to poor selection of measurement matrix, too much reduction in sample size, or large error variance.

The compressive sampling technique is described in this paper because it is a similar data compression procedure which selects a few linear combinations of the observations. The present work is neither a development to the CS theory nor it is claimed to uniformly outperform the existing CS algorithms. The comparison using the simulated data only indicates that ASREM and ESREM provide reasonable estimation. In terms of accuracy, ESREM performs comparably with existing CS algorithm for moderate error level, but requires less computation time. For larger error levels, it outperforms the existing CS algorithm.

Our approach can be extended to a non-iid setup where the observations may be generated from a stochastic process. A multivariate extension can also be done where we can do a two-fold reduction in terms of dimension and sample size assuming samples from a multivariate distribution.

3.12. Appendix

3.12.1. Detailed calculations of the proposed approaches: The derivations and the detailed expression of the M-steps of ASREM and ESREM are described here.

3.12.1.1. *Calculation of the M-step in ASREM:.* The conditional distribution of $x|y, \mu^{(t)}$ is given by

$$N_n\big(\mu^{(t)} + \phi'(\phi\phi^{'})^{-1}(y - \phi\mu^{(t)}),\ \sigma^2(I_n - \phi'(\phi\phi^{'})^{-1}\phi)\big). \tag{3.12.1}$$

Now $\mu \in S_i$ implies certain linear restrictions on the parameter μ in the form

$$A_i \mu = 0$$

where $A_i = ((a^i_{\gamma\delta}))$ is a $n \times n$ matrix such that

$$a^i_{\delta\delta} = \begin{cases} 1 & \text{if}\, \mu_\delta = 0 \\ \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad a^i_{\gamma\delta} = 0 \quad \text{if}\, \gamma \neq \delta.$$

Using Lagrange multiplier method to incorporate the restrictions we construct

$$Q_R(\mu) = Q(\mu) - \lambda^{'} A_i \mu$$

where $\lambda^{'} = (\lambda_1, \lambda_2, \ldots, \lambda_n)$ are Lagrangean multipliers.

We set $\frac{\partial}{\partial \mu_j} Q_R(\mu) = 0$ for all j. If $\mu_j \neq 0$

$$\frac{\partial}{\partial \mu_j} Q_R(\mu) = 0 \implies \frac{\partial}{\partial \mu_j} Q(\mu) = 0$$

$$\implies \frac{\partial}{\partial \mu_j} \big[E\{ \sum_{\alpha=1}^{n} (x_\alpha - \mu_\alpha)^2 | y, \mu^{(t)} \} \big] = 0$$

$$\implies \hat{\mu}_j^{(t+1)}(S_i) = E[x_j | y, \mu^{(t)}]$$

$$\implies \hat{\mu}_j^{(t+1)}(S_i) = \mu_j^{(t)} + \alpha_j - \beta_j$$

where $\alpha_j = j^{th}$ element of $\phi(\phi\phi^{'})^{-1} y$ and $\beta_j = j^{th}$ element of $\phi^{'}(\phi\phi^{'})^{-1}\phi\mu^{(t)}$ (from (3.12.1))

If $\mu_j = 0$

$$\frac{\partial}{\partial \mu_j} Q_R(\mu) = 0$$

$$\implies \hat{\mu}_j^{(t+1)}(S_i) = E[x_j | y, \mu^{(t)}] - \lambda_j$$

From the restriction $\mu_j = 0$ we get

$$\lambda_j = E[x_j|y, \mu^{(t)}]$$

$$\Longrightarrow \hat{\mu}_j^{(t+1)}(S_i) = 0.$$

Then we choose the $\hat{\mu}^{(t+1)}(S_i)$ for which $Q(\hat{\mu}^{(t+1)}(S_i))$ is maximum as the new estimate of μ at $(t+1)^{th}$ iteration.Thus the estimate of μ is

$$\hat{\mu}^{(t+1)} = \hat{\mu}^{(t+1)}(S_i)$$

such that

$$Q(\hat{\mu}^{(t+1)}(S_i)) \geq Q(\hat{\mu}^{(t+1)}(S_j)) \; \forall j = 1, 2, \cdots, n.$$

3.12.1.2. *Calculation regarding in ESREM:.* For the unrestricted EM algorithm the estimate of $\boldsymbol{\mu}$ should converge to the maximizer of the observed log-likelihood. The equation for finding the MLE from the observed likelihood is

$$(\phi' V^{-1} \phi)\mu = \phi' V^{-1} y \tag{3.12.2}$$

where $V = \phi\phi'$.

Now the above equation (3.12.2) does not have a unique solution as $rank[(\phi' V^{-1}\phi)_{n\times n}] = m \ll n$. Hence the observed likelihood does not have a unique maximum and our unrestricted EM algorithm will produce many estimates of μ. Among these many estimates we choose the sparsest solution. This is taken care of by taking the initial estimate of μ as 0 in the iterative process as then the estimate will hopefully converge to nearest solution which will be the sparsest one. We have justify this with the help of simulation in section 9. For finding the least norm solution of (3.12.2) we take the Moore-Penrose inverse of $(\phi^{'}V^{-1}\phi)$ [**33**] and find the unrestricted EM estimate as

$$\hat{\mu}^{un} = (\phi^{'}V^{-1}\phi)^{+}\phi^{'}V^{-1}y = Py$$

where $P = (\phi^{'}V^{-1}\phi)^{+}\phi^{'}V^{-1}$.

Then the distribution of $\hat{\mu}^{un}$ comes out to be $N_n(P\phi\mu\ ,\ \sigma^2 PVP^{'})$, so that we get

$$E(\hat{\mu}^{un}) = P\phi\mu = (\phi^{'}V^{-1}\phi)^{+}(\phi^{'}V^{-1}\phi)\mu.$$

Since μ is sparse we may assume $(\phi^{'}V^{-1}\phi)^{+}(\phi^{'}V^{-1}\phi)\mu \simeq \mu$ and hence we get $E(\hat{\mu}^{un}) = P\phi\mu \simeq \mu$.

Now for identifying the true subspace we want to test n hypotheses

$$H_{0i} : \mu_i = 0 \qquad \forall i = 1(1)n$$

Let

$$\hat{\mu}^{un} = (\hat{\mu}_1^{un}, \hat{\mu}_2^{un}, \ldots \hat{\mu}_n^{un})^{'}$$

The above calculations shows that the test statistics for testing H_{0i} is

$$\tau_i = |\frac{\hat{\mu}_i^{un}}{\sigma\sqrt{s_{ii}}}| \sim N(0,1) \quad \text{under } H_{0i} \qquad \forall i = 1(1)n$$

where $s_{ii} = i^{th}$diagonal element of $PVP^{'}$. This justifies the choice of the estimated subspace in 3.6.1.

3.12.2. Proofs of the theoretical properties. Below we give the proofs of theorem 2 and theorem 3. The theorems are stated earlier at section 7.

3.12.2.1. *Proof of Theorem 23:*

PROOF. First we consider the ASREM algorithm.We apply the REM for each of the subspaces S_i in each iteration. The only modification at this stage is that in the M-step of each iteration we choose the maximum of the Q values over different subspaces. At the $(t+1)^{th}$ iteration in ASREM we have

$$Q(\hat{\mu}^{(t+1)}(S_i)) \geq Q(\hat{\mu}^{(t+1)}(S_j)) \, \forall j \neq i$$

$$\Rightarrow Q(\hat{\mu}^{(t+1)}) = \max_j Q(\hat{\mu}^{(t+1)}(S_j)).$$

Also since we apply REM for each of the subspace S_i, it follows from the property of REM that for S_i we have

$$Q(\hat{\mu}^{(t+1)}(S_i)) \geq Q(\hat{\mu}^{(t)}(S_i)) \, \forall i.$$

Hence we have the following chain of inequalities

$$Q(\hat{\mu}^{(t+1)}) = \max_j Q(\hat{\mu}^{(t+1)}(S_j)) \geq \max_j Q(\hat{\mu}^{(t)}(S_j)) = Q(\hat{\mu}^{(t)})$$

$$\Rightarrow \ell_{obs}(\hat{\mu}^{(t+1)}) \geq \ell_{obs}(\hat{\mu}^{(t)}).$$

□

Thus for ASREM we get that the Q function is nondecreasing in each iteration which makes the observed log-likelihood also nondecreasing in each iteration. Next we consider the ESREM algorithm. Here we apply the REM over the estimated subspace $\hat{S}_\mu$. Hence from the property of REM we find that the observed log-likelihood is nondecreasing with each iteration.

3.12.2.2. *Proof of Theorem 24:*

PROOF. The insignificant components of $\hat{\mu}^{un}$ are set to zero and the subspace spanned by the basis corresponding to the significant components is chosen. We find the restricted estimate $\hat{\mu}$ by maximizing $Q(\mu)$ over this subspace. Thus the restricted estimate $\hat{\mu}$ can be treated as a projection of $\hat{\mu}^{un}$ on $\hat{S}_\mu$. Hence if $\mu \in \hat{S}_\mu$ we have

$$\| \hat{\mu} - \mu \|_{l_2} \leq \| \hat{\mu}^{un} - \mu \|_{l_2} \qquad w.p.\ 1.$$

This implies

$$E[\| \hat{\mu} - \mu \|_{l_2}] \leq E[\| \hat{\mu}^{un} - \mu \|_{l_2}]$$

Now

$$\| \hat{\mu}^{un} - \mu \|_{l_2}^2 = \sum_{i=1}^{n} (\hat{\mu}_i^{un} - \mu_i)^2$$

$$\Rightarrow E[\| \hat{\mu}^{un} - \mu \|_{l_2}^2] = \sum_{i=1}^{n} E(\hat{\mu}_i^{un} - \mu_i)^2 = \sum_{i=1}^{n} V(\hat{\mu}_i^{un}) \qquad [\because E(\hat{\mu}_i^{un}) = \mu_i]$$

$$\Rightarrow E[\| \hat{\mu}^{un} - \mu \|_{l_2}^2] = \sigma^2 \sum_{i=1}^{n} s_{ii}.$$

Then

$$E^2[\| \hat{\mu}^{un} - \mu \|_{l_2}] \leq E[\| \hat{\mu}^{un} - \mu \|_{l_2}^2] = \sigma^2 \sum_{i=1}^{n} s_{ii}$$

$$\Rightarrow E[\| \hat{\mu}^{un} - \mu \|_{l_2}] \leq \sigma \sqrt{\sum_{i=1}^{n} s_{ii}} \qquad [\because\ s_{ii} > 0\ \forall i].$$

This implies

$$E[\parallel \hat{\mu} - \mu \parallel_{l_2}] \leq E[\parallel \hat{\mu}^{un} - \mu \parallel_{l_2}] \leq \sigma \sqrt{\sum_{i=1}^{n} s_{ii}}$$

$$\Rightarrow E[\parallel \hat{\mu} - \mu \parallel_{l_2}] \leq \sigma \sqrt{\sum_{i=1}^{n} s_{ii}}.$$

□

CHAPTER 4

Data Reduction in Markov model

4.1. Introduction

In this chapter we shall consider the problem of data reduction in case of dependent data setup. More specifically here we shall discuss filtering mechanism in case of discrete Markov models. Discrete Markov chains are the simplest dependent structure that one can think of and are very useful for modeling a wide range of scientific problems in nature. Some important applications include modeling of dry and wet spells (P. J. Avery and D. A. Henderson (1999) [**5**]), deoxyribonucleic acid (DNA) sequences (P. J. Avery and D. A. Henderson (1999) [**5**]), study of chronic diseases (B. A. Craig and A. A. Sendi (2002) [**17**]).

Any stochastic process $\mathbf{X} = \{X_1, X_2, \cdots, X_n\}$ having a finite set $\mathcal{S}$ as its state space, is said to be a Markov process of order s if

$$P(X_n = a_n|\, X_{n-1} = a_{n-1}, X_{n-2} = a_{n-2}, \cdots, X_1 = a_1)$$

$$= P(X_n = a_n|\, X_{n-1} = a_{n-1}, X_{n-2} = a_{n-2}, \cdots, X_{n-s} = a_{n-s})$$

For notational convenience let us denote the state space as $\mathcal{S}$ =$\{1, 2, \cdots, k\}$. Further we assume that the Markov process has stationary transition probabilities, which means

$$P(X_n = a_n|\, X_{n-1} = a_{n-1}, X_{n-2} = a_{n-2}, \cdots, X_{n-s} = a_{n-s}) = p_{a_{n-s},\cdots,a_{n-1}:a_n}$$

does not depend on n. For $s = 1$ we have a simple Markov chain with finite state space. Any Markov chain of s^{th} order can be treated as a simple Markov chain with suitable parameters. Hence in this chapter we shall develop the methods assuming a simple Markov chain which will be equally applicable for any Markov process of higher order. A Markov chain can be completely described by the initial state and the set of transition probabilities. Here we shall consider the initial state of a Markov chain to be known and try to make inferences about the transition probabilities based on the observed data. More specifically inferences regarding the transition probability matrix can help us to answer many specific questions regarding the Markov process which we usually encounter.

There is an extensive literature available on the statistical inferences of finite Markov chains based on complete data. Billingsley [**10**] gives a good account of the mathematical aspects of different techniques regarding inferences about the transition probabilities which includes Whittle's formula, maximum likelihood and chi-square methods. Estimation of transition probabilities and testing goodness of fit from a single realisation of a Markov chain has been studied by Bartlett [**9**]. Goodman and Anderson [**4**] derived the estimates of the transition probabilities and their hypothesis when there are more than one realisation of a single Markov chain. Their paper also described the asymptotic properties of the methods when the number of realisations increase. All these works assume the observed data to be one or more long, unbroken observations of the chain. In this paper we assume that there is a single realisation of the Markov chain which is not completely observed. The observed broken chain which results from the filtering mechanism is therefore not Markov.

Based on the filtering mechanism we will observe only certain transitions of a Markov chain and treat the remaining part of the chain as missing. From the observed data we will estimate the transition probabilities using EM algorithm. Since the missingness in the data occurs due to the filtering process, the data are not missing at random (NMAR) and the missing mechanism is nonignorable but known. The E-step of the EM algorithm requires to find the conditional expectation of the missing data given the observed data. This is achieved by defining the all possible missing paths for a transition of any order and finding the probability of the same. The standard error of the EM estimate is obtained by the supplemented EM algorithm (SEM) (Meng and Rubin [**39**]). Usually the standard error of the EM estimate is obtained by inverting the observed information matrix. In our case the observed likelihood cannot be obtained explicitly and hence we avoid the calculation of the observed information matrix. SEM is a technique to calculate asymptotic dispersion matrix of the EM estimate without inverting the observed information matrix.

Section 2 describes the setup of the problem. Section 3 deals with the identifiability issues of the parameter that arise due to filtering of data. In section 2 we assume that the transition probability matrix consists of all positive elements. This assumption is relaxed in section 3 where we allow some structural zeroes in the transition probability matrix. We describe the additional modification we need in the filtering mechanism due to such relaxation. Section 5 describes the methods of estimation and testing the transition probabilities. In this section we also describe the estimation of standard errors of the estimates by the SEM algorithm. Section 6 describes the generalization of the above methods in case of multiple Markov chains. In section 7 we demonstrate the methods developed using simulated data. A real life data analysis is demonstrated in section 8. Section 9 is the appendix which has the proofs of a major theorem of this paper.

4.2. Setup

Let X be a simple Markov chain with finite state space $\mathcal{S} = \{1, 2, \cdots, k\}$ and transition probability matrix $P = ((p_{ij}))_{k \times k}$. Let us first assume that $0 < p_{ij} < 1$, $\forall\, i, j$. We shall relax this assumption later and consider the case where we allow some p_{ij}'s to be zero. The transition probability matrix P satisfy the standard condition

$$P1 = 1 \; i.e. \; \sum_j p_{ij} = 1 \text{ for all } i.$$

Hence there are $k^2 - k$ independent parameters. We define the vector of the parameters as

$$\theta = (p_{11}, p_{12}, \cdots, p_{1(k-1)}, p_{21}, p_{22}, \cdots, p_{2(k-1)}, \cdots, p_{k1}, p_{k2}, \cdots, p_{k(k-1)})'$$

$$= (\theta_1, \theta_2, \cdots, \theta_d)'$$

where $d = k^2 - k$ and the parameter space is

$$\Theta = \Big\{\mu : \sum_{j=1}^{k-1} p_{ij} < 1, \quad \text{for } i = 1, \cdots, k\Big\}$$

$$= \Big\{\theta : \sum_{j=1}^{k-1} \theta_j < 1\ , \sum_{j=0}^{k-1} \theta_{(i-1)k+j} < 1, \quad \text{for } i = 2, \cdots, k\Big\}.$$

We consider a single realization x of the chain and the number of transitions from state i to state j in this realization is n_{ij}. We assume that the Markov process is continued sufficiently long enough so that the realization x contains each transition at least once, that is, $n_{ij} > 0$ for all i and j. The matrix of transition count is

$$N = \begin{bmatrix} n_{11} & n_{12} & \cdots & \cdots & n_{1k} \\ n_{21} & n_{22} & \cdots & \cdots & n_{2k} \\ & & \vdots & & \\ n_{\overline{k-1}1} & n_{\overline{k-1}2} & \cdots & \cdots & n_{\overline{k-1}k} \\ n_{k1} & n_{k2} & \cdots & \cdots & n_{kk} \end{bmatrix}.$$

In this chapter we shall propose a data acquisition protocol which suggests that instead of observing the entire realization x, we record only some of the transitions and treat the remaining part of the chain as missing. The decision about which transitions we record is described in the form of a filter matrix $F = ((f_{ij}))_{1\leq i,j\leq k}$ which contains 0 and 1 as elements. In particular we record the transition from state i to state j if $f_{ij} = 1$. If X is the complete chain then let $\phi_F(X)$ denote the chain filtered using **F**. So in conjunction with our general notation, the filtering mechanism here can be described in the form of a matrix and the filtered data is $Y = \phi_F(X)$.

EXAMPLE 25. Consider a three state Markov chain x as

$$112312232123331121331$$

Suppose we are given a filter matrix

$$F = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}.$$

Then the transitions we record in $\phi_F(x)$ are

$$\begin{gathered} 1 \to 1 \\ 2 \to 2 \\ 3 \to 1 \\ 3 \to 2 \end{gathered} \tag{4.2.1}$$

Then the filtered chain is

$$11\textvisiblespace 312232\textvisiblespace\textvisiblespace\textvisiblespace\textvisiblespace 311\textvisiblespace\textvisiblespace\textvisiblespace 31$$

In the filtered chain the missing states are indicated by the symbol "␣" which we call "blank". The example shows that besides the transitions (4.2.1) there may be some transitions which are indirectly recorded in the filtered chain (such as $2 \to 3$ is recorded even if $f_{23} = 0$). Any transition $i \to j$ may be recorded indirectly in the filtered chain if there exist some states a and b such that $f_{ai} = 1$ and $f_{jb} = 1$. Thus all the transitions in the filtered chain may be classified into one of the three categories:

- directly recorded ($f_{ij} = 1$)
- indirectly recorded ($f_{ij} = 0$ but the transition occurs in the filtered chain, e.g. $2 \to 3$ in Example 25)
- unobserved ($f_{ij} = 0$ and the transition does not appear in the filtered chain e.g. $3 \to 3$ in Example 25)

4.3. Identifiability

In this section we shall discuss about the identifiability of the parameters based on the filtered chain. We note that our filtered chain no longer possesses the Markov property and hence the issue of identifiability needs to be studied separately. While applying the filtering mechanism, if we record only a very few transitions then all the parameters of the Markov chain may not be identifiable. For example in a Markov chain with state space $\{1, 2, \cdots, 10\}$ if we record only the transition $1 \to 1$ then some parameters , say p_{55}, are not identifiable. We need to study how much data we can throw away, so that the problem still remains identifiable. Thus our main aim, in this section, will be to identify a class of filter matrices so that data generated by any filter matrix of that class will retain the identifiability of the parameters. But first we define what is meant by identifiability of parameter on the basis of a random sample.

Definition 26. Let X be a random sample from a distribution characterized by the parameter θ and $L(\theta, x)$ be the likelihood. Then the parameter θ is said to be identifiable on the basis of X if for any two distinct values θ_1 and θ_2 in the parameter space

$$L(\theta_1, x) \neq L(\theta_2, x).$$

Suppose X is a random sample drawn from a population characterized by the parameter θ. Let $Y = g(X)$ be function of X. Given X we can always construct Y through g. So if θ is identifiable on the basis of Y, we can identify θ also from X. On the contrary, if θ is unidentifiable on the basis of X, then it is also unidentifiable on the basis of Y. This is because, if we assume θ to be identifiable on the basis of Y, then given X, we can construct Y through g and then θ can be identified from X, which is a contradiction. Thus in general we have the following two results:

Claim 27. a) If θ is identifiable on the basis of Y, then θ is also identifiable on the basis of X.

b) If θ is unidentifiable on the basis of X, then θ is also unidentifiable on the basis of Y.

In the present situation to prove that the parameters are identifiable it is enough to consider a observed sample x such that $\exists t$ such that $P_\theta(\phi_F(x) = t) \neq P_{\theta'}(\phi_F(x) = t)$ and prove that any two different values of the parameter θ will yield different values of the observed likelihood $L_{obs}(\theta, x)$.

Let $\mathcal{F}$ be the class of all $k \times k$ filter matrices. We call a filter matrix $F \in \mathcal{F}$ identifiable if P is identifiable with respect to $\phi_F(X)$.

Let $\mathcal{I}_\theta \subseteq \mathcal{F}$ be the set of all $k \times k$ filter matrices for which the parameter θ is identifiable. Then $\mathcal{I} = \cap \mathcal{I}_\theta$ is the set of identifiable filter matrices.

With this notation, the general fact stated in claim 27 is also applicable for the data generated by the filter matrices.

LEMMA 28. *For $H, M \in \mathcal{F}$, let $\phi_H = g \circ \phi_M$ for some function $g(.)$. Then $H \in \mathcal{I}$ implies $M \in \mathcal{I}$ and $M \in \mathcal{F} - \mathcal{I}$ implies $H \in \mathcal{F} - \mathcal{I}$.*

EXAMPLE 29. Let

$$H = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix} \qquad \text{and} \qquad M = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}.$$

M is same as H except that for M we directly observe one more transition $2 \to 3$ than H. Then $\phi_H = g \circ \phi_M$ and hence if $H \in \mathcal{I}$ then $M \in \mathcal{I}$.

In general there are 2^{k^2}possible filter matrices in $\mathcal{F}$. Instead of searching over all possible filter matrices we shall start with some definite structures of filter matrices which are identifiable. The above discussion motivates us to extend the identifiability over a larger class of matrices. This requires some ordering of the filter matrices in terms of the data we store.

DEFINITION 30. For filter matrix $M = ((m_{ij})) \in \mathcal{F}$ and $H = ((h_{ij})) \in \mathcal{F}$ we say $M \succeq H$ if $\forall i, j \quad h_{ij} = 1 \Rightarrow m_{ij} = 1$ and $M \preceq H$ if $\forall i, j \quad h_{ij} = 0 \Rightarrow m_{ij} = 0$.

LEMMA 31. *a) If $H \in \mathcal{I}$ and $M \succeq H$ then $M \in \mathcal{I}$.*

b) If $H \in \mathcal{F} - \mathcal{I}$ and $M \preceq H$ then $M \in \mathcal{F} - \mathcal{I}$.

PROOF. a) $M \succeq H$ implies $\phi_H = g(\phi_M)$ for some $g(.)$. Using Lemma 28 we get $H \in \mathcal{I}$ implies $M \in \mathcal{I}$.

b) $M \preceq H$ implies $\phi_M = g(\phi_H)$ for some $g(.)$. Using Lemma 28 we get $H \in \mathcal{F} - \mathcal{I}$ implies $M \in \mathcal{F} - \mathcal{I}$. □

Thus if any filter matrix M is identifiable, then all filter matrices which stores more data than M are also identifiable. This fact is also true for any subclass of filter matrices.

DEFINITION 32. If $\mathcal{D} \subseteq \mathcal{F}$, then the closure of $\mathcal{D}$ is defined as

$$\bar{\mathcal{D}} = \{F \in \mathcal{F} \ : F \succeq D \text{ for some } D \in \mathcal{D}\}.$$

LEMMA 33. *If $\mathcal{D} \subseteq \mathcal{I}$ then $\bar{\mathcal{D}} \subseteq \mathcal{I}$.*

PROOF. Let $M \in \bar{\mathcal{D}}$. Then $M \succeq D$ for some $D \in \mathcal{D}$.
Since $\mathcal{D} \subseteq \mathcal{I}$, we get $D \in \mathcal{I}$. Then Lemma 31 implies $M \in \mathcal{I}$.
This implies $\bar{\mathcal{D}} \subseteq \mathcal{I}$. □

Thus given any class of identifiable filter matrices $\mathcal{D}$ we can always extend it to a larger subclass of identifiable filter matrices.

Our observed chain is a sequence of states and blanks (␣). Given any observed chain we want to find the condition under which the conditional probability of a given segment of the observed chain given the initial state in the segment is identifiable.

Definition 34. For any finite sequence π of states or blanks ($\_\_\_$) we define

$$S_\pi = \text{set of all filtered segments where } \pi \text{ occurs in consecutive positions.}$$

We note that if $\pi_1 \subseteq \pi_2$ then $S_{\pi_1} \supseteq S_{\pi_2}$.

Lemma 35. *For any filter matrix F, if $P(S_\pi) > 0$ then p_π is identifiable where π is a sequence of states or blanks which starts and ends with states and p_π is the conditional probability of the sequence π given the initial state in π.*

Proof. Let π start with the state α and end with the state β.

Let

$$\mathcal{A} = \text{subchains that ends with } \alpha.$$

$$\mathcal{B} = \text{subchains that ends with the sequence } \pi.$$

Then $\mathcal{B} \subseteq \mathcal{A}$. Also $P(S_\pi) > 0$ implies $P(\mathcal{B}) > 0$ which implies $P(\mathcal{A}) > 0$.

Also from Markov property we get that $P(\mathcal{B}|\mathcal{A}) = p_\pi$. Thus if p_π changes $P(\mathcal{B}|\mathcal{A})$ changes. Since the conditional probability of a class of subchains changes, the joint distribution of the entire filtered chain must also change. Hence two distinct values of p_π will give two distinct values of the observed likelihood.Thus p_π is identifiable. □

Corollary 36. *For any filter matrix F the parameter p_{ij} is identifiable if $P(S_{ij}) > 0$.*

As mentioned before we want to start with filter matrices of definite structures which are identifiable and extend them to relatively larger classes. With this view in mind, we define three classes of filter matrices each of which will be sufficient for a filter matrix to be identifiable.

Class1: We define $\mathbb{C}_1 \subseteq \mathcal{F}$ which consists of all filter matrix $F = ((f_{ij}))$, $1 \leq i, j \leq k$ such that

a) $\exists$ α such that $f_{\alpha j} = 0, j = 1, 2, ..., k$ i.e. the α^{th} row of F is zero.

b) $\exists$ β such that $f_{i\beta} = 0, i = 1, 2, ..., k$ i.e. the β^{th}column of F is zero.

c) $f_{pj} = 1$ for exactly one j, $1 \leq j \leq k$, $\quad p = 1, 2, ..., k$, $p \neq \alpha$,i.e. except α^{th} row every other row must have exactly one element 1.

d) $f_{iq} = 1$ for exactly one i, $1 \leq i \leq k$, $\quad q = 1, 2, ..., k$, $q \neq \beta$,i.e. except β^{th} column every other column must have exactly one element 1.

Class2: We define $\mathbb{C}_2 \subseteq \mathcal{F}$ which consists of all filter matrix $F = ((f_{ij}))$, $1 \leq i, j \leq k$ such that

a) $\exists$ α and β such that $f_{i\alpha} = 0, i = 1, 2, ..., k$ and $f_{i\beta} = 0, i = 1, 2, ..., k$ i.e. the α^{th} and β^{th} column of F is zero.

b) $f_{iq} = 1$ for at exactly one i, $1 \leq i \leq k$, $\quad q = 1, 2, ..., k$, $q \neq \alpha, \beta$,i.e. except α^{th} and β^{th} column every other column have exactly one element 1.

c) $f_{\alpha j} = f_{\beta j} = 1 \quad 1 \leq j \leq k, \qquad , j \neq \alpha, \beta$, i.e. except α^{th} and β^{th} column every other element of α^{th} and β^{th} row is 1.

d) $f_{pj} = 1$ for exactly one j, $1 \leq j \leq k, \qquad p = 1, 2, ..., k, p \neq \alpha, \beta$,i.e. except α^{th} and β^{th} row every other row have exactly one element 1.

Class3: We define $\mathbb{C}_3 \subseteq \mathcal{F}$ which consists of all filter matrix $F = ((f_{ij}))$, $1 \leq i, j \leq k$ such that

a) $\exists$ α and β such that $f_{\alpha i} = 0, i = 1, 2, ..., k$ and $f_{\beta i} = 0, i = 1, 2, ..., k$ i.e. the α^{th} and the β^{th} row of F is zero.

b) $f_{qi} = 1$ for exactly one i, $1 \leq i \leq k, \qquad q = 1, 2, ..., k$, $q \neq \alpha, \beta$,i.e. except α^{th} and β^{th} row every other row have exactly one element 1.

c) $f_{j\alpha} = f_{j\beta} = 1 \quad 1 \leq j \leq k, \qquad , j \neq \alpha, \beta$, i.e. except α^{th} and β^{th} row every other element of α^{th} and β^{th} column is 1.

d) $f_{jp} = 1$ for exactly one j, $1 \leq j \leq k, \qquad p = 1, 2, ..., k,$ $p \neq \alpha, \beta$,i.e. except α^{th} and β^{th} column every other column have exactly one element 1.

The following theorem and its corollary provide sufficient conditions for filter matrices to be identifiable. Any filter matrix which belong to at least one of the three classes is identifiable. The proof of the theorem is given in the appendix.

THEOREM 37. *Consider an univariate Markov chain X on finite state space $\{1, 2, ..., k\}$ and transition probabilities p_{ij} where $0 < p_{ij} <$*

$1, i, j = 1, 2, \ldots k$. *Suppose F be any filter matrix belonging to the class $\mathbb{C}_* = \mathbb{C}_1 \cup \mathbb{C}_2 \cup \mathbb{C}_3$. Then F must also belong to the class $\mathcal{I}$.*

The following corollary to the above theorem is an immediate application of Lemma 33.

Corollary 38. $\overline{\mathbb{C}_*} \subseteq \mathcal{I}$.

Thus if we start with a definite structure of matrices in $\mathbb{C}_1$ or $\mathbb{C}_2$ or $\mathbb{C}_3$ we get a relatively larger class $\overline{\mathbb{C}_*}$ of identifiable filter matrices. For the rest of the paper we shall be working with filter matrices within this class. We shall find that any filter matrix in this class will provide considerable reduction in data.

4.4. Structural zeroes in Transition probability matrix

In the previous section while obtaining the sufficient conditions for identifiability we assumed $0 < p_{ij} < 1$, $\forall\, i, j$. This was a crucial assumption in developing the theory for the sufficient conditions. However in many practical applications this assumption stands out to be too restrictive. For example, while modeling a disease status the probability of an individual entering from one state to another may be zero (in case of chronic illness, the condition of an individual usually deteriorates). Also the case of structural zeroes in the transition probability matrix will occur later in this paper while dealing with multiple Markov chains. In this section we generalize the sufficient conditions for a filter matrix to be identifiable even when some p_{ij}'s are zero.

We note that all zeroes (if any) in the transition probability model are structural zeroes, that is, we know the position of the zeroes even before the collection of the data. Also for any i, p_{ij} must be positive for at least one j since all the row sums of the transition probability matrix is 1. We further assume

(**A1**) for any j, p_{ij} must be positive for at least one i.

This is a reasonable assumption to make because if such a state j exists we shall ignore that state from our analysis.

As before we have the classes of filter matrix $\mathbb{C}_1$, $\mathbb{C}_2$ and $\mathbb{C}_3$. Further let us define an additional class of filter matrix $\mathfrak{R} \subseteq \mathcal{F}$ as

$$\mathfrak{R} = \{F \in \mathcal{F} : \text{ For any } i \in \{1, 2, ..., n\}, f_{ij} = 1 \text{ for at least one } j \in Z\}$$

where $Z = \{j : p_{ij} > 0\}$. This restriction means for every row of a filter matrix, we should observe at least one probable transition. The restriction on the filter matrices is quite justified and does not in any way reduces the applicability of filtering mechanism. The following theorem is a generalization of Theorem 37 in the case where we allow some p_{ij} to be zero.

THEOREM 39. *Consider an univariate Markov chain X on finite state space $\{1, 2, ..., k\}$ and transition probabilities p_{ij} where $0 \leq p_{ij} \leq 1, i, j = 1, 2, ...k$. Let F be any filter matrix belonging to the class $\overline{\mathcal{S}}$ where $\mathcal{S} = \mathbb{C}_* \cap \mathfrak{R}$. Then under the assumption A1, F must also belong to the class $\mathcal{I}$.*

The proof of the above theorem is similar to the proof of Theorem 37 because under the assumption A1, and for filter matrices within the class $\mathcal{S}$, we have $P(S_\pi) > 0$ for all choices of sequences π, that we require in Theorem 37. Finally application of lemma 33, gives the required result.

4.5. Estimation and testing

As mentioned earlier, a Markov process can be completely characterized by specifying the transition probability matrix. This section deals with drawing inferences regarding the parameters. Instead of recording the entire Markov chain x, we apply a given filter matrix $F \in \mathcal{F}$ to record $\phi_F(x)$. F is fixed and does not in any way depend on the data x. The choice of F may depend on the availability of the samples, storage facilities or past experience subject to the constraint of identifiability. Based on $\phi_F(x)$ we shall find estimates of the transition probabilities and compute the standard error of the estimates. Our main tool for estimation will be EM algorithm. For the computation of the standard error we shall use Supplemented EM algorithm(SEM). The latter part of the section deals with testing of hypothesis regarding the parameters.

4.5.1. Estimation of parameters: In the present situation the complete data is x which is unobserved and the observed data is $\phi_F(x)$. As a natural tool of missing data analysis we will apply EM algorithm for the estimation of the parameter θ. Each iteration of EM algorithm consists of a E-step (expectation step) and an M-step (maximization step). In the E-step of the algorithm we need to find the conditional expectation of the complete data log-likelihood given the observed data and the current iterated value of the parameters. In our case this requires to find the conditional expectation with respect to the conditional distribution of x given $\phi_F(x)$ and the current iterated value $\theta^{(t)}$ of the parameter. The complete data log likelihood is

$$\ell_{com}(\theta) = \text{constant} + \sum_{\alpha,\beta=1}^{k} n_{\alpha\beta} \log p_{\alpha\beta}.$$

Since $\ell_{com}(\theta)$ is linear in $n_{\alpha\beta}$, we need to compute

$$E\Big(n_{\alpha\beta}|\phi_F(x);\theta^{(t)}\Big).$$

This conditional distribution cannot be computed directly as the conditional distribution of x given $\phi_F(x)$ cannot be found out explicitly. We shall express $n_{\alpha\beta}$ as a sum of certain indicator variables to evaluate this conditional expectation, the computation of which will be shown in subsection 4.5.1.2. We shall show that this require us to find the conditional probability that the observed chain $\phi_F(x)$ moves from state α to state β as

$$P_{\theta^{(t)}}\Big(\phi_F(x)_i = \beta|\phi_F(x)_{i-1} = \alpha\Big)$$

where $\phi_F(x)_k$ is the k^{th} value of the observed chain $\phi_F(x)$. Since the observed chain has runs of missing states, the calculation of the above probability will require us to find the probability of a transition from one state to another in any number of steps such that all the intermediate steps are missing. If the complete chain is available, then the probability of a transition from a to b in ν steps is the $(a,b)^{th}$ element of

P^{ν}. However we need to find the probability of such transition through some specific ways.

4.5.1.1. *Defining possible missing paths for a transition:* Consider two states a and b. Suppose we are interested in transition from a to b in ν steps. Each possible way of transition from a to b in ν steps is called a path of order ν. We call a path of order 1 as edge. Thus any given path consists of one or more edges. Clearly the transition from a to b in ν steps can occur through one or more paths. We classify these paths in two categories based on the given filter matrix:

- **observed path**($\mathcal{O}$): whose all edges are observed.
- **unobserved path**($\mathcal{U}$): whose all edges are unobserved.

Clearly the two sets $\mathcal{O}$ and $\mathcal{U}$ are not mutually exhaustive, that is, we cannot classify all paths into any one of these categories.

Example 40. Consider a two state Markov chain and two filter matrices F_1 and F_2 such that

$$F_1 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \qquad F_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Suppose we consider the *transition from state* 1 *to state* 1 *in two steps.* The possible paths are:

$$w_1 : 1 \longrightarrow 1 \longrightarrow 1 \qquad w_2 : 1 \longrightarrow 2 \longrightarrow 1$$

For filter matrix F_1, path $w_1 \in \mathcal{O}$, i.e. path $w_1$is observed whereas $\mathcal{U}$ is empty, i.e. no paths are unobserved. For filter matrix F_2, path $w_1 \in \mathcal{O}$ and $w_2 \in \mathcal{U}$. If we consider the *transition from state* 2 *to state* 2 *in two* *steps*, the possible paths are:

$$w_1 : \ 2 \longrightarrow 2 \longrightarrow 2 \qquad\qquad w_2 : \ 2 \longrightarrow 1 \longrightarrow 2$$

For filter matrix F_1, path $w_1 \in \mathcal{O}$, and $\mathcal{U}$ is empty whereas for filter matrix F_2, path $w_1 \in \mathcal{O}$ and $w_2 \in \mathcal{U}$.

Now consider the transition probability matrix P of the Markov chain. We construct two matrices $P^{[0]} = ((p_{ij}^{[0]}))$ and $P^{[1]} = ((p_{ij}^{[1]}))$ as

$$p_{ij}^{[0]} = \begin{cases} 0 & \text{if } f_{ij} = 1 \\ p_{ij} & \text{if } f_{ij} = 0 \end{cases}$$

and

$$p_{ij}^{[1]} = \begin{cases} 0 & \text{if } f_{ij} = 0 \\ p_{ij} & \text{if } f_{ij} = 1 \end{cases}$$

Then the $(i,j)^{th}$ element of $(P^{[0]})^{\nu}$ gives the probability of going *from state i to state j in ν steps through unobserved path(s).* Also the $(i,j)^{th}$ element of $(P^{[1]})^{\nu}$ gives the probability of going *from state i to state j in ν steps through observed path(s).*

Example 41. (Example 40 continued) Returning to the previous example we see that for the filter matrix F_1,

$$P = \begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{bmatrix} \qquad P^{[0]} = \begin{bmatrix} 0 & p_{12} \\ 0 & 0 \end{bmatrix} \qquad P^{[1]} = \begin{bmatrix} p_{11} & 0 \\ p_{21} & p_{22} \end{bmatrix}.$$

Then

$$(P^{[0]})^2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad (P^{[1]})^2 = \begin{bmatrix} p_{11}^2 & 0 \\ p_{21}p_{11} + p_{22}p_{21} & p_{22}^2 \end{bmatrix}.$$

Thus for filter matrix F_1, probability of going from any state i to any state j through the unobserved paths in 2 steps is zero. Also

$$(P^{[0]})^\nu = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \text{for any } \nu$$

which means that the probability of going from any state i to any state j through the unobserved paths in any steps is zero. Similarly for filter matrix F_2,

$$P^{[0]} = \begin{bmatrix} 0 & p_{12} \\ p_{21} & 0 \end{bmatrix} \qquad (P^{[0]})^2 = \begin{bmatrix} p_{12}p_{21} & 0 \\ 0 & p_{21}p_{12} \end{bmatrix}$$

Thus for F_2, the probability of going from state 1 to state 1 in 2 steps through the unobserved paths is $p_{12}p_{21}$ and the probability of going from state 2 to state 2 in 2 steps through the unobserved paths is $p_{21}p_{12}$.

Thus given a filter matrix, the probability of going from a state a to a state b in ν steps through the unobserved paths is the $(a, b)^{th}$ element of $(P^{[0]})^\nu$ which is $p_{ab}^{(\nu)0}$.

4.5.1.2. *Estimation by EM Algorithm:* For the i^{th} transition, let,

$$Y_{1i} = \text{state from where the transition occurs}$$

$$Y_{2i} = \text{state to where the transition occurs}$$

Thus Y_{1i} and Y_{2i} are two discrete random variables taking values in the state space $\{1, 2, \cdots k\}$ for all i. Let us express the total number of transitions $n_{\alpha\beta}$ from the state α to the state β as

$$n_{\alpha\beta} = \sum_{i=1}^{n} I(Y_{1i} = \alpha, Y_{2i} = \beta)$$

where

$$I(Y_{1i} = \alpha, Y_{2i} = \beta) = \begin{cases} 1 & if\ Y_{1i} = \alpha, Y_{2i} = \beta \\ 0 & \text{otherwise.} \end{cases}.$$

The complete data likelihood then can be written as

$$L_{com}(p) \propto \prod_{i=1}^{n} f(y_{1i}, y_{2i}|p) = constant \times \prod_{\alpha,\beta=1}^{k} p_{\alpha\beta}^{n_{\alpha\beta}}$$

where

$$p_{\alpha k} = 1 - \sum_{j=1}^{k-1} p_{\alpha j}\ \forall\alpha = 1(1)n.$$

After t iterations in the EM algorithm we write the E-step and the M-step as follows:

E-step:

Let $P_{(t)} = ((p_{\alpha\beta(t)}))$ be the value of the transition probability matrix after t iterations. The corresponding value of the parameter θ is $\theta^{(t)}$. The other matrices we construct take the values $P_{(t)}^{[0]}$ and $(P_{(t)}^{[0]})^{\nu} = ((p_{ab(t)}^{(\nu)0}))$. Then we compute the expected complete data log-likelihood with respect to the conditional distribution of $x|\phi_F(x), \theta^{(t)}$. The complete data log-likelihood is given by

$$\ell_{com}(\theta) = constant + \sum_{\alpha,\beta=1}^{k} \Big\{ (\log p_{\alpha\beta}) \times n_{\alpha\beta} \Big\}.$$

We then compute

$$Q(\theta) = E(\ell_{com}(\theta)|\phi_F(x), \theta^{(t)}).$$

Since $\ell_{com}(\theta)$ is linear in $n_{\alpha\beta}$, we need to compute

$$\begin{aligned} & E\Big(n_{\alpha\beta}|\phi_F(x), \theta^{(t)}\Big) \\ = & \sum_{i=1}^{n} E\Big(I(Y_{1i} = \alpha, Y_{2i} = \beta)|\phi_F(x), \theta^{(t)}\Big) \\ = & \sum_{i=1}^{n} P\Big(Y_{1i} = \alpha, Y_{2i} = \beta|\phi_F(x), \theta^{(t)}\Big). \end{aligned}$$

Let us denote $P\Big(Y_{1i} = \alpha, Y_{2i} = \beta|\phi_F(x), \theta^{(t)}\Big) = P^i_{\alpha\beta}$. Then for each i, $P^i_{\alpha\beta}$ takes one of the three forms $P^{i(1)}_{\alpha\beta}$, $P^{i(2)}_{\alpha\beta}$ or $P^{i(3)}_{\alpha\beta}$ as follows:

- Case I (Y_{1i} observed): Suppose we have a missing chain of length $\nu - 1$ with the next observed state b. Then

$$P^i_{\alpha\beta} = \begin{cases} \frac{p_{\alpha\beta} \times p^{(\nu-1)0}_{\beta b}}{p^{(\nu)0}_{\alpha b}} & \text{if } Y_{1i} = \alpha \\ 0 & \text{if } Y_{1i} \neq \alpha \end{cases} =: P^{i(1)}_{\alpha\beta}, \text{ say.}$$

- Case II (Y_{2i} observed): Suppose we have a missing chain of length $\nu - 1$ with the previous observed state a. Then

$$P^i_{\alpha\beta} = \begin{cases} \frac{p^{(\nu-1)0}_{a\alpha} \times p_{\alpha\beta}}{p^{(\nu)0}_{a\beta}} & \text{if } Y_{2i} = \beta \\ 0 & \text{if } Y_{2i} \neq \beta \end{cases} =: P^{i(2)}_{\alpha\beta}, \text{ say.}$$

- Case III (Both are not observed): Suppose we have a missing chain of length $\nu - 1$ with the previous observed state a and the next observed state b. Then

$$P^i_{\alpha\beta} = \frac{p^{(m)0}_{a\alpha} p_{\alpha\beta} p^{(n)0}_{\beta b}}{p^{(\nu)0}_{ab}} =: P^{i(3)}_{\alpha\beta}, \text{ say.}$$

  where $m + n = \nu - 1$ and $a = Y_{1(i-m+1)}$ and $b = Y_{2(i+n)}$. If there is no such next observed state (that is, the observed chain ends) then

$$P^i_{\alpha\beta} = \frac{p^{(m)0}_{a\alpha} p_{\alpha\beta} (\sum\limits_b p^{(n)0}_{\beta b})}{\sum\limits_b p^{(\nu)0}_{ab}} =: P^{i(3)}_{\alpha\beta}, \text{ say.}$$

M-step:

We try to maximize $Q(\theta)$ with respect to θ. Setting $\frac{\partial}{\partial \theta_j} Q(\theta) = 0$ for each $j = 1(1)d$ we get

$$\theta^{(t+1)} = (\theta^{(t+1)}_1, \theta^{(t+1)}_2, \cdots, \theta^{(t+1)}_d)$$

where

$$\theta_j^{(t+1)} = \frac{\displaystyle\sum_{l=1}^{n} P_{1j}^l}{\displaystyle\sum_{\beta=1}^{k}\sum_{l=1}^{n} P_{1\beta}^l} \quad \text{for any } j = 1, 2, \cdots, (k-1)$$

and

$$\theta_{(i-1)k+j}^{(t+1)} = \frac{\displaystyle\sum_{l=1}^{n} P_{i(j+1)}^l}{\displaystyle\sum_{\beta=1}^{k}\sum_{l=1}^{n} P_{i\beta}^l} \quad \text{for any } j = 0, 1, \cdots, (k{-}1) \text{ and } i = 2, 3, \cdots, k.$$

4.5.2. Estimation of Standard Errors: Since EM estimates of the parameters are the maximum likelihood estimate of the observed likelihood, the large sample covariance matrix can be obtained by inverting the observed information matrix. But in our problem the observed likelihood is not known explicitly. An alternative way is using Supplemented EM Algorithm (SEM) by Meng and Rubin [**39**] which allows us to find the large sample covariance matrix without inverting the estimate of the observed information matrix. SEM algorithm is a procedure of obtaining a numerically stable estimate of the covariance matrix of the estimated parameters using only the code for the steps in EM algorithm, code for computing the large sample complete data covariance matrix and standard matrix operations.

4.5.2.1. *Supplemented EM Algorithm:* Since each step of the EM algorithm produces a fresh estimate of the parameter from the previous estimates, EM algorithm can be considered as a mapping $\mathcal{M}$ on the parameter space. The derivative of the EM mapping, which we denote as $\mathcal{M}_{(1)}$, can be expressed in the form

$$\mathcal{M}_{(1)} = i_{mis} i_{com}^{-1} = I - i_{obs} i_{com}^{-1}.$$

The above equation implies

$$i_{obs}^{-1} = i_{com}^{-1}(I - \mathcal{M}_{(1)})^{-1}$$

which in turn implies

$$V_{obs} = V_{com}(I - \mathcal{M}_{(1)})^{-1}.$$

Now we note that

$$V_{obs} = V_{com}(I + \mathcal{M}_{(1)} - \mathcal{M}_{(1)})(I - \mathcal{M}_{(1)})^{-1} = V_{com} + \triangle\, V$$

where $\triangle\, V = V_{com} M_{(1)}(I - \mathcal{M}_{(1)})^{-1}$ is the increment in variance due to missingness.

4.5.2.2. *Calculation of V_{com}:* The complete data log-likelihood is given by

$$\ell_{com}(\theta) = constant + \sum_{i=1} n_{ij} \log p_{ij} \qquad \text{where } p_{ik} = 1 - \sum_{j=1}^{k-1} p_{ij}\ \forall i$$

so that

$$\frac{\partial}{\partial p_{ij}} \ell_{com} = \frac{n_{ij}}{p_{ij}} - \frac{n_{ik}}{p_{ik}}.$$

Thus the gradient vector is

$$\mathbf{S} = \begin{bmatrix} \frac{n_{11}}{p_{11}} - \frac{n_{1k}}{p_{1k}} \\ \vdots \\ \frac{n_{k(k-1)}}{p_{k(k-1)}} - \frac{n_{kk}}{p_{kk}} \end{bmatrix}.$$

Now for any $i \neq i'$,we have $\frac{\partial^2}{\partial p_{ij} \partial p_{i'j'}} \ell_{com} = 0$. Also

$$\frac{\partial^2}{\partial p_{ij} \partial p_{ij'}} \ell_{com} = \begin{cases} \frac{n_{ik}}{2p_{ik}^2} & \text{if } j \neq j' \\ \frac{1}{2}\left[\frac{n_{ik}}{p_{ik}^2} - \frac{n_{ij}}{p_{ij}^2}\right] & \text{if } j = j' \end{cases}.$$

Let B be the matrix of the negatives of the second order derivatives. Then B is a matrix of order $k^2 - k$ such that

$$B = blockdiagonal(B_1, B_2, \cdots B_k)$$

where $B_i = ((b^i_{jj'}))_{k-1}$ and

$$b^i_{jj'} = -\frac{\partial^2}{\partial p_{ij} \partial p_{ij'}} \ell_{com}.$$

Then the fisher information matrix of the complete data is

$$i_{com} = E(B\,|\theta, data) = blockdiagonal\Big(E(B_1), E(B_2), \cdots, E(B_k)\Big)$$

where $E(B_i) = ((E(b^i_{jj'}|\theta, data)))$. Thus the variance-covariance matrix of the complete data is $V_{com} = i^{-1}_{com}$.

4.5.2.3. *Computing $\mathcal{M}_{(1)}$ by numerical differentiation:* For our problem the mapping $\mathcal{M} = M(\theta_1, \theta_2, \cdots, \theta_d) : \Theta \to \Theta$ is not known explicitly. The derivative of $\mathcal{M}$ at $\hat{\theta}$ is calculated numerically from the output

Algorithm 1 SEM Algorithm

```
We take as input $\hat{\theta}$and $\theta^{(t)}$.
a) Run the usual E-step and M-steps to get $\theta^{(t+1)}$;
b) Fix $i = 1$. Calculate
       $$\theta^{(t)}(i) = (\hat{\theta}_1, \cdots \hat{\theta}_{i-1}, \theta_i^{(t)}, \hat{\theta}_{i+1}, \cdots, \hat{\theta}_d)$$
   which is $\hat{\theta}$ except the $i^{th}$ component which equals $\theta_i^{(t)}$.
c) Treating $\theta^{(t)}(i)$ as the current estimate of $\theta$, run one iteration
   of EM to obtain $\tilde{\theta}^{(t+1)}(i)$.
d) Obtain the ratio
       $$r_{ij}^{(t)} = \frac{\tilde{\theta}_j^{(t+1)}(i) - \hat{\theta}_j}{\theta_i^{(t)} - \hat{\theta}_i} \qquad \text{for } j = 1, 2, \cdots, d.$$
e) Repeat steps 2 to 4 for $i = 1, 2, \cdots, d$.
   We get as output $\theta^{(t+1)}$ and $\{r_{ij}^{(t)} : i, j = 1, 2, \cdots, d\}$.
   $M_{(1)}$ is the limiting matrix $\{r_{ij}\}$ as $t \to \infty$.
```

of the forced EM steps. $\mathcal{M}_{(1)}$ is the matrix with the $(i, j)^{th}$element as

$$\frac{\triangle \mathcal{M}_j}{\triangle \hat{\theta}_i} = \text{change in the } j^{th} \text{ component of } \mathcal{M} \text{ due to the change in the } i^{th}\text{element of } \hat{\theta}.$$

For this we start with the EM estimate $\hat{\theta}$ and change its i^{th} element $\hat{\theta}_i$ by $\theta_i^{(t)}$. We call this resultant estimate by $\theta^{(t)}(i)$ and run one EM iteration on it to get $\tilde{\theta}^{(t+1)}(i)$. Then

$$\triangle \mathcal{M}_j = \tilde{\theta}_j^{(t+1)}(i) - \hat{\theta}_j$$

and

$$\triangle \hat{\theta}_i = \theta_i^{(t)} - \hat{\theta}_i$$

and so we compute the ratio $r_{ij} = \frac{\triangle \mathcal{M}_j}{\triangle \hat{\theta}_i}$. Thus we run a sequence of SEM iterations, where the $(t+1)^{th}$ iteration is defined as in the algorithm 1.

A difficulty in running the SEM iterations is that while changing the i^{th} element $\hat{\theta}_i$ by $\theta_i^{(t)}$ the resultant estimate $\theta^{(t)}(i)$ may not belong to the parameter space Θ because the sum of the corresponding row probabilities $\sum_{j=1}^{k-1} p_{ij}$ may be more than 1. Thus theoretically the mapping M may not be defined in such cases. Then we replace $\theta_i^{(t)}$ by $\theta_i^{(t)} - \epsilon$, $(\epsilon > 0)$ so that the corresponding sum of probability is less than 1.

4.5.2.4. *Implementational Issues:* While implementing the SEM algorithm it is always safe to start with the initial values of the original EM algorithm for numerical accuracy. But this may result in too many unnecessary iterations because the initial choice may be too far from the MLE. Hence Meng and Rubin suggested to take the initial choice in SEM as a suitable iterate of the EM algorithm or two complete data standard deviations from the MLE. Computation of $\mathcal{M}_{(1)}$ being numerical differentiation is less accurate than evaluating the function $\mathcal{M}$ itself. Hence the stopping criterion should be less stringent for SEM algorithm as compared to the original EM algorithm. Meng and Rubin suggested to use square root of the stopping criterion of the original EM as the stopping criterion for SEM.

The observed variance covariance matrix obtained by SEM algorithm should be theoretically a real symmetric positive definite matrix. This provide a diagnostics for programming errors and numerical precision. The numerical symmetry of the final matrix increases with more stringent criterion in the algorithm.

4.5.3. Testing of Hypotheses. The large sample inferences on the EM estimate can be drawn using the asymptotic distribution

$$\hat{\theta} \sim N(\theta, V_{obs}).$$

Since SEM algorithm helps us to numerically estimate V_{obs}, we can use the above distribution for testing of the parameters and finding confidence intervals.

4.5.3.1. *Testing the transition probability matrix.* Suppose we wish to test the hypothesis $H_0 : P = P_0$. Since only $k(k-1)$ parameters of the transition probability matrix are independent, the above hypothesis is equivalent to $H_0 : \theta = \theta_0$. Now

$$(\hat{\theta} - \theta)' V_{obs}^{-1} (\hat{\theta} - \theta) \sim \chi^2_{k^2}$$

asymptotically which implies the test statistic for testing H_0 is $\chi^2 = (\hat{\theta} - \theta_0)' V_{obs}^{-1} (\hat{\theta} - \theta_0)$ which has asymptotically $\chi^2_{k^2}$ distribution under H_0. Thus the critical region for testing H_0 is $\{x : \chi^2 > \chi^2_{k^2,\alpha}\}$

4.5.3.2. *Test of hypotheses about specific probabilities and confidence regions.* First we consider testing the hypothesis that certain transition probabilities p_{ij} have specified values p^0_{ij}. Under the null hypothesis $H_{0i} : \theta_i = \theta^0_i$, the statistic $\tau_i = \frac{\hat{\theta}_i - \theta^0_i}{\sqrt{s_{ii}}}$ has $N(0,1)$ distribution. Thus the critical region for testing H_{0i} is $\{x : |\tau_i| > z_{\alpha/2}\}$. The $100(1-\alpha)\%$ confidence interval for θ_i is

$$(\hat{\theta}_i - \sqrt{s_{ii}} z_{\alpha/2}, \hat{\theta}_i + \sqrt{s_{ii}} z_{\alpha/2}).$$

4.6. Multiple Markov chains

Let $\{X_n\}$ be a s^{th} order Markov chain. In the previous sections we have discussed the case where $s = 1$, that is, simple Markov chains. If $s > 1$, then $\{X_n\}$ is called a multiple Markov chain of order s with transition probabilities

$$p_{a_1,\cdots,a_s:a_{s+1}} = P(X_n = a_{s+1}|\, X_{n-1} = a_s, X_{n-2} = a_{s-1}, \cdots, X_{n-s} = a_1).$$

Multiple Markov chains of any order can be reduced to a simple Markov chain by the following technique.

Suppose $\{X_n\}$ is called a Markov chain of order s with k states. We define a new stochastic process $\{Y_n, n = 1, 2, \cdots\}$ where $Y_n = (X_n, X_{n+1}, \cdots, X_{n+s-1})$. Then $\{Y_n\}$ is a simple Markov chain whose state space has k^s different $s-$tuples. The transition probabilities of the new defined Markov process are

$$p_{(a_1,a_2,\cdots,a_s)(b_1,b_2,\cdots,b_s)} = \begin{cases} p_{a_1 a_2 \cdots a_s:b_s} & \text{if } \ b_i = a_{i+1}, i = 1, 2, \cdots, s-1 \\ 0 & \text{otherwise.} \end{cases}$$

The number of positive entries in the $k^s \times k^s$ transition probability matrix is k^{s+1}. The parameters of interest are the probabilities $p_{a_1,\cdots,a_s:a_{s+1}}$ which requires estimation from the data.

In this situation we apply our filtering technique to the chain $\{y_n\}$. But now the transition probability matrix contains many zero elements and hence the additional restriction described section 4 needs to be applied on the filter matrices. We note that in this case the transition probability matrix satisfies the assumption made in section 5. The technique of estimation of the parameters from the data $\phi_F(y)$ remains same as in the simple Markov chain.

4.7. Simulation Study

For simulation we start with a Markov chain with 3 states. A Markov chain of length 1000 is being generated with the transition probability matrix

$$P = \begin{bmatrix} 0.2 & 0.3 & 0.5 \\ 0.8 & 0.1 & 0.1 \\ 0.7 & 0.1 & 0.2 \end{bmatrix}.$$

The filter matrix for generating the observed chain is

$$F = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

Clearly the filter matrix used satisfy the sufficient condition for estimability. With this filter matrix we reduce 16% of the data, i.e., from the complete Markov chain of length 1000 we do not observe 16% of the data. The precision we use in estimating the parameters through the steps of the EM algorithm is of the order 10^{-12} and the precision used in computing the standard error is of the order 10^{-6}. With this precision the estimated transition probability matrix is

$$\hat{P} = \begin{bmatrix} 0.2411168 & 0.2850831 & 0.4738001 \\ 0.7395851 & 0.1429865 & 0.1174284 \\ 0.7648870 & 0.1067367 & 0.1283763 \end{bmatrix}.$$

The observed variance covariance matrix V_{obs} as computed by the SEM algorithm is

$$\begin{bmatrix} 8.00 \times 10^{-4} & -3.00 \times 10^{-4} & 2.36 \times 10^{-5} & 4.55 \times 10^{-6} & 7.13 \times 10^{-4} & 7.78 \times 10^{-5} \\ -3.00 \times 10^{-4} & 4.70 \times 10^{-4} & -8.85 \times 10^{-6} & -1.71 \times 10^{-6} & -2.68 \times 10^{-4} & -2.92 \times 10^{-5} \\ 2.36 \times 10^{-5} & -8.84 \times 10^{-6} & 1.01 \times 10^{-3} & -5.10 \times 10^{-4} & -8.05 \times 10^{-7} & -5.30 \times 10^{-5} \\ 4.60 \times 10^{-6} & -1.73 \times 10^{-6} & -5.10 \times 10^{-4} & 6.06 \times 10^{-4} & -5.92 \times 10^{-8} & -1.02 \times 10^{-5} \\ 7.13 \times 10^{-4} & -2.68 \times 10^{-4} & -7.52 \times 10^{-7} & -1.45 \times 10^{-7} & 1.80 \times 10^{-3} & -1.06 \times 10^{-4} \\ 7.78 \times 10^{-5} & -2.92 \times 10^{-5} & -5.30 \times 10^{-5} & -1.02 \times 10^{-5} & -1.06 \times 10^{-4} & 3.92 \times 10^{-4} \end{bmatrix}.$$

The complete data variance covariance matrix V_{com} is

$$\begin{bmatrix} 0.0003674 & -0.0001380 & & & & \\ -0.0001380 & 0.0004092 & & & & \\ & & 0.0009496 & -0.0005214 & & \\ & & -0.0005214 & 0.0006041 & & \\ & & & & 0.0006033 & -0.0002739 \\ & & & & -0.0002739 & 0.0003199 \end{bmatrix}$$

The increase in variance $\triangle V$ is

$$\begin{bmatrix} 4.33 \times 10^{-4} & -1.63 \times 10^{-4} & 2.36 \times 10^{-5} & 4.58 \times 10^{-6} & 7.13 \times 10^{-4} & 7.78 \times 10^{-5} \\ -1.63 \times 10^{-4} & 6.11 \times 10^{-5} & -8.86 \times 10^{-6} & -1.71 \times 10^{-6} & -2.68 \times 10^{-4} & -2.92 \times 10^{-5} \\ 2.35 \times 10^{-5} & -8.84 \times 10^{-6} & 5.75 \times 10^{-5} & 1.11 \times 10^{-5} & -8.06 \times 10^{-7} & -5.30 \times 10^{-5} \\ 4.61 \times 10^{-6} & -1.73 \times 10^{-6} & 1.11 \times 10^{-5} & 2.15 \times 10^{-6} & -5.92 \times 10^{-8} & -1.02 \times 10^{-5} \\ 7.13 \times 10^{-4} & -2.68 \times 10^{-4} & -7.52 \times 10^{-7} & -1.45 \times 10^{-7} & 1.20 \times 10^{-3} & 1.68 \times 10^{-4} \\ 7.78 \times 10^{-5} & -2.92 \times 10^{-5} & -5.30 \times 10^{-5} & -1.02 \times 10^{-5} & 1.68 \times 10^{-4} & 7.22 \times 10^{-5} \end{bmatrix}$$

4.8. Practical example

The data consists of the daily rainfall, measured in millimeters times 10, at Alofi in the Niue Island group. 1096 observations were recorded from 1st January 1987 until 31st December 1989. The data is classified into three states: state 1 which represents "no rain", state 2 which represents "from non zero until 5mm" and state 3 which represents "more than 5mm" rain. This time series data can be considered as a 3 state Markov chain. P. J. Avery and D. A. Henderson (1999) [**5**] used this dataset for the fitting of Markov model.

For the generation of the observed data we use the same filter matrix as in case of the simulated data. From 1096 observations we find that this filter matrix leads to a missingness of 45.35%. While storing only 54.65% of the original data we find the estimate of the transition probability matrix is

$$\begin{bmatrix} 0.6717154 & 0.2231926 & 0.1050920 \\ 0.4585938 & 0.3034812 & 0.2379251 \\ 0.2137608 & 0.3447883 & 0.4414509 \end{bmatrix}.$$

We compute the observed variance covariance matrix as

$$\begin{bmatrix} 4.65\times10^{-4} & -3.16\times10^{-4} & 1.23\times10^{-6} & 8.14\times10^{-7} & 1.15\times10^{-4} & 1.81\times10^{-4} \\ -3.16\times10^{-4} & 3.41\times10^{-4} & -8.51\times10^{-7} & -5.63\times10^{-7} & -7.76\times10^{-5} & -1.23\times10^{-4} \\ 1.23\times10^{-6} & -8.51\times10^{-7} & 9.21\times10^{-4} & -4.18\times10^{-4} & -4.76\times10^{-5} & -3.09\times10^{-4} \\ 8.14\times10^{-7} & -5.63\times10^{-7} & -4.18\times10^{-4} & 7.49\times10^{-4} & -3.15\times10^{-5} & -2.05\times10^{-4} \\ 1.15\times10^{-4} & -7.76\times10^{-5} & -4.76\times10^{-5} & -3.15\times10^{-5} & 9.26\times10^{-4} & 1.51\times10^{-4} \\ 1.81\times10^{-4} & -1.23\times10^{-4} & -3.09\times10^{-4} & -2.05\times10^{-4} & 1.51\times10^{-4} & 0.0026 \end{bmatrix}.$$

The complete data variance covariance matrix V_{com} is

$$\begin{bmatrix} 0.000391 & -0.000266 & & & & \\ -0.000266 & 0.000307 & & & & \\ & & 0.000837 & -0.000469 & & \\ & & -0.000469 & 0.0007128 & & \\ & & & & 0.0007185 & -0.000315 \\ & & & & -0.000315 & 0.0009658 \end{bmatrix}.$$

The increase in variance due to missingness is

$$\begin{bmatrix} 7.48\times10^{-5} & -5.00\times10^{-5} & 1.23\times10^{-6} & 8.14\times10^{-7} & 1.15\times10^{-4} & 1.81\times10^{-4} \\ -5.00\times10^{-5} & 3.38\times10^{-5} & -8.50\times10^{-7} & -5.63\times10^{-7} & -7.76\times10^{-5} & -1.23\times10^{-4} \\ 1.23\times10^{-6} & -8.50\times10^{-7} & 8.39\times10^{-5} & 5.55\times10^{-5} & -4.76\times10^{-5} & -3.09\times10^{-4} \\ 8.14\times10^{-7} & -5.63\times10^{-7} & 5.55\times10^{-5} & 3.68\times10^{-5} & -3.15\times10^{-5} & -2.05\times10^{-4} \\ 1.15\times10^{-4} & -7.76\times10^{-5} & -4.76\times10^{-5} & -3.15\times10^{-5} & 2.08\times10^{-5} & 4.66\times10^{-4} \\ 1.81\times10^{-4} & -1.23\times10^{-4} & -3.09\times10^{-4} & -2.05\times10^{-4} & 4.66\times10^{-4} & 0.0016079 \end{bmatrix}.$$

4.9. Appendix

Proof of Theorem 37:

Proof. We split the proof in three parts. We shall prove $\mathbb{C}_i \subseteq \mathcal{I}, i = 1, 2, 3$. This will imply that $\mathbb{C}_* \subseteq \mathcal{I}$.

Part 1:

Suppose a filter matrix $M \in \mathbb{C}_1$. Then the α^{th} row and β^{th}column of M are zero and all other rows and columns of M have exactly one element nonzero.

Case a: $\alpha \neq \beta$

Step 1:

Consider $p_{ij} \quad 1 \leq i, j \leq k \ \ , i, j \neq \alpha, \beta$.

Let the i^{th}column has a element $f_{ai} = 1$ and let the j^{th} row has a element $f_{jb} = 1$.

Then $P(S_{aijb}) > 0$. This implies $P(S_{ij}) > 0$.

Hence corollary 12 implies that $p_{ij} \quad 1 \leq i, j \leq k \ \ , i, j \neq \alpha, \beta$ are estimable.

Step 2:

Next from the β^{th} column of the transition probability matrix consider $p_{i\beta}, \forall i = 1(1)k,\ i \neq \beta$.

Since $\beta \neq \alpha$, we have a j such that $f_{\beta j} = 1$

Also since $i \neq \beta$ we have a such that $f_{ai} = 1$

Then $P(S_{ai\beta j}) > 0$. This implies $P(S_{i\beta}) > 0$.

Again corollary 12 implies that $p_{i\beta}, \forall i = 1(1)k,\ i \neq \beta$ are estimable.

Step 3:

Next from the α^{th} row of the transition probability matrix consider $p_{\alpha j}, \forall j = 1(1)k,\ j \neq \alpha$.

For $p_{\alpha j}$ choose i and r such that $f_{i\alpha} = 1 \quad i \neq \alpha$ and $f_{jr} = 1$

Then $P(S_{i\alpha jr}) > 0$. This implies $P(S_{\alpha j}) > 0$.

From corollary 12 we get $p_{\alpha j}, \forall j = 1(1)k,\ j \neq \alpha$ is estimable.

Step 4:

The parameter $p_{\alpha\alpha}$ is estimable from the condition $\sum\limits_{j} p_{\alpha j} = 1$

Step 5:

From the β^{th} row of the transition probability matrix consider $p_{\beta j}, \forall j = 1(1)k,\ j \neq \alpha$.

If j is such that $f_{\beta j} = 1$ then we get that $p_{\beta j}$ is estimable. Hence we now consider j to be such that $f_{\beta j} = 0$.

For this we now choose any state a and a state s such that $f_{js} = 1$.

Then $P(S_{a\_js}) > 0$ which implies $P(S_\pi) > 0$ where $\pi = a\_js$.

Let $C = \{b : f_{ab} = 0 \quad , f_{bj} = 0\}$. We note that $\beta \in C$ and p_π is of the form

$$p_\pi = (\sum_{b \in C, b \neq \beta} p_{ab}p_{bj} + p_{a\beta}p_{\beta j}) \times p_{js}$$

Now since $P(S_\pi) > 0$, lemma 11 implies that p_π is identifiable. Hence

$$p_\pi = (\sum_{b \in C, b \neq \beta} p_{ab}p_{bj} + p_{a\beta}p_{\beta j}) \times p_{js} = \text{Known Constant}$$

Since all p_{ab} and p_{bj} and also $p_{a\beta}$ are identifiable , we get $p_{\beta j}, \forall j = 1(1)k,\ j \neq \alpha$ are estimable.

Step 6:

From the α^{th} column of the transition probability matrix consider $p_{i\alpha}, \forall i = 1(1)k,\ i \neq \beta$

If i is such that $f_{i\alpha} = 1$ then we get that $p_{i\alpha}$ is estimable. Hence we now consider i to be such that $f_{i\alpha} = 0$.

For this we now choose any state b and a state r such that $f_{ri} = 1$

Then $P(S_{ri\_b}) > 0$ which implies $P(S_\pi) > 0$ where $\pi = ri\_b$.

Let $D = \{a : f_{ab} = 0 \quad , f_{ia} = 0\}$. We note that $\alpha \in D$ and p_π is of the form

$$p_\pi = p_{ri} \times (\sum_{a \in D, a \neq \alpha} p_{ia} p_{ab} + p_{i\alpha} p_{\alpha b})$$

Now since $P(S_\pi) > 0$, lemma 11 implies that p_π is identifiable. Hence

$$p_\pi = p_{ri} \times (\sum_{a \in D, a \neq \alpha} p_{ia} p_{ab} + p_{i\alpha} p_{\alpha b}) = \text{Known Constant}$$

Since all p_{ab} and p_{ia} and also $p_{\alpha b}$ are identifiable , we get $p_{i\alpha}, \forall i = 1(1)k,\ i \neq \beta$ are estimable.

Step 7:

$p_{\beta\alpha}$ is estimable from the condition $\sum_j p_{\beta j} = 1$

Case b : $\alpha = \beta$

Step 1:

Consider $p_{ij} \quad 1 \leq i, j \leq k \ \ , i, j \neq \alpha$.

The estimatibility of p_{ij} is same as step 1 of **case (a).**

Step 2:

Next from the α^{th} column of the transition probability matrix consider $p_{i\alpha}, \forall i = 1(1)k,\ i \neq \alpha$.

The parameter $p_{i\alpha}$ is identified from the condition $\sum_j p_{ij} = 1$

Step 3:

Next from the α^{th} row of the transition probability matrix consider $p_{\alpha j}, \forall j = 1(1)k,\ j \neq \alpha$

For $p_{\alpha j}$ choose i and r such that $f_{i\alpha} = 0 \quad i \neq \alpha$ and $f_{jr} = 1$.

Then $P(S_{i\_jr}) > 0$ which implies $P(S_\pi) > 0$ where $\pi = i\_jr$.

Let $D = \{b : f_{bj} = 0 \quad \text{and} \quad f_{ib} = 0\}$. We note that $\alpha \in D$ and p_π is of the form

$$p_\pi = (\sum_{b \in D, b \neq \alpha} p_{ib}p_{bj} + p_{i\alpha}p_{\alpha j})p_{jr}$$

Now since $P(S_\pi) > 0$ lemma 11 implies that p_π is identifiable. Hence

$$p_\pi = (\sum_{b \in D, b \neq \alpha} p_{ib}p_{bj} + p_{i\alpha}p_{\alpha j})p_{jr} = \text{Known Constant}$$

Since each of $p_{ib}, p_{bj}, p_{i\alpha}$ in the above equation are already identifiable, we get that $p_{\alpha j}$ can also be identified uniquely.

Step 4 :

The parameter $p_{\alpha\alpha}$ is identified from the condition $\sum\limits_j p_{\alpha j} = 1$

Thus all the parameters for M are identifiable. Hence for any matrix $M \in \mathbb{C}_1$, we have $M \in \mathcal{F}$. Thus $\mathbb{C}_1 \subseteq \mathcal{I}$.

Part 2:

In the next case, suppose a filter matrix $M \in \mathbb{C}_2$. Then the α^{th} and β^{th}column of a filter matrix M are zero and all other columns of M have exactly one element nonzero.

Step 1:

Consider $p_{ij} \quad 1 \leq i, j \leq k \ \ , i \neq \alpha, \beta$.

Let the i^{th} column has a element $f_{ai} = 1$ and let the j^{th} row has a element $f_{jr} = 1$.

Then $P(S_{aijr}) > 0$. This implies $P(S_{ij}) > 0$.

Hence applying the corollary 12 $p_{ij} \quad 1 \leq i, j \leq k \ \ , i \neq \alpha, \beta$ are estimable.

Step 2:

Consider $p_{\alpha\alpha}$ and $p_{\beta\alpha}$.

Let the α^{th} row has a element $f_{\alpha r} = 1$, $r \neq \alpha, \beta$ and we choose a i such that $i \neq \alpha, \beta$.

Then $P(S_{i\_\alpha r}) > 0$ which means $P(S_\pi) > 0$ where $\pi = i \_ \alpha r$.

Let $D = \{b : f_{ib} = 0 \quad \text{and} \quad f_{b\alpha} = 0\}$. Then p_π is of the form

$$p_\pi = (\sum_{b \in D} p_{ib} p_{b\alpha}) p_{\alpha r}$$

Clearly $\alpha, \beta \in D$ and hence we get

$$p_\pi = (\sum_{b \in D, b \neq \alpha, \beta} p_{ib} p_{b\alpha} + p_{i\alpha} p_{\alpha\alpha} + p_{i\beta} p_{\beta\alpha}) p_{\alpha r}$$

Since $P(S_\pi) > 0$, lemma 8 implies that p_π is identifiable. Hence

$$p_\pi = (\sum_{b \in D, b \neq \alpha, \beta} p_{ib} p_{b\alpha} + p_{i\alpha} p_{\alpha\alpha} + p_{i\beta} p_{\beta\alpha}) p_{\alpha r} = \text{Known Constant}$$

Since $p_{ib}, b \neq \alpha$ and $p_{b\alpha}, b \neq \alpha, \beta$ and $p_{\alpha r}, r \neq \alpha, \beta$ are all estimable from the above equation we get a equation of the form

This gives us a equation of the form

$$C_1 p_{\alpha\alpha} + C_2 p_{\beta\alpha} = K_1$$

where $C_i^{\prime}s$ are constants.

Also we from the condition $\sum_j p_{ij} = 1$, since all other parameters are estimable we get a equation of the form

$$p_{\alpha\alpha} + p_{\beta\alpha} = K_2$$

These two equations make $p_{\alpha\alpha}$and $p_{\beta\alpha}$ estimable.

Step 3:

Consider $p_{\alpha\beta}$ and $p_{\beta\beta}$.

Let the β^{th} row has a element $f_{\beta r} = 1$, $r \neq \alpha, \beta$ and we choose a i such that $i \neq \alpha, \beta$.

Then $P(S_{i\ \_\ \beta r}) > 0$ which means $P(S_\pi) > 0$ where $\pi = i\ \_\ \beta r$.

Let $D = \{b : f_{ib} = 0 \quad \text{and} \quad f_{b\beta} = 0\}$. Then p_π is of the form

$$p_\pi = (\sum_{b \in D} p_{ib} p_{b\beta}) p_{\beta r}$$

Clearly $\alpha, \beta \in D$ and hence we get

$$p_\pi = (\sum_{b \in D, b \neq \alpha, \beta} p_{ib} p_{b\beta} + p_{i\alpha} p_{\alpha\beta} + p_{i\beta} p_{\beta\beta}) p_{\beta r}$$

Since $P(S_\pi) > 0$, lemma 11 implies that p_π is identifiable. Hence

$$p_\pi = (\sum_{b \in D, b \neq \alpha, \beta} p_{ib} p_{b\beta} + p_{i\alpha} p_{\alpha\beta} + p_{i\beta} p_{\beta\beta}) p_{\beta r} = \text{Known Constant}$$

Since $p_{ib}, b \neq \alpha$ and $p_{b\beta}, b \neq \alpha, \beta$ and $p_{\beta r}, r \neq \alpha, \beta$ are all estimable from the above equation we get a equation of the form

$$C_1 p_{\alpha\beta} + C_2 p_{\beta\beta} = K_1$$

where C_1 and C_2 and K_1 are constants.

Also we from the condition $\sum_j p_{ij} = 1$, since all other parameters are estimable we get a equation of the form

$$p_{\alpha\beta} + p_{\beta\beta} = K_2$$

These two equations make $p_{\alpha\beta}$and $p_{\beta\beta}$ estimable.

Thus all the parameters for M are identifiable. Hence for any matrix $M \in \mathbb{C}_2$, we have $M \in \mathcal{F}$. Thus $\mathbb{C}_2 \subseteq \mathcal{I}$.

Part 3:

Now suppose a filter matrix $M \in \mathbb{C}_3$. Then the α^{th} and β^{th}row of a filter matrix M are zero and all other rows of M have exactly one element nonzero.

Step 1:

Consider $p_{ij} \quad 1 \leq i,j \leq k \ \ ,j \neq \alpha,\beta$.

Let the i^{th} column has a element $f_{ai} = 1$ and let the j^{th} row has a element $f_{jr} = 1$.

Then $P(S_{aijr}) > 0$. This implies $P(S_{ij}) > 0$.

Hence applying the corollary 12 $p_{ij} \quad 1 \leq i,j \leq k \ \ ,j \neq \alpha,\beta$ are estimable.

Step 2:

Consider $p_{\alpha j} \ \ j = \alpha,\beta$.

Let the α^{th} column has a element $f_{i\alpha} = 1\, ,i \neq \alpha,\beta$ and we choose a r such that $r \neq \alpha,\beta$.

Then $P(S_{i\alpha\_r}) > 0$ which means $P(S_{\pi}) > 0$ where $\pi = i\alpha\_r$.

Let $D = \{b : f_{\alpha b} = 0 \quad \text{and} \quad f_{br} = 0\}$. Then p_{π} is of the form

$$p_{\pi} = p_{i\alpha}(\sum_{b \in D} p_{\alpha b}p_{br})$$

Clearly $\alpha,\beta \in D$ and hence we get

$$p_{\pi} = (\sum_{b \in D, b \neq \alpha,\beta} p_{\alpha b}p_{br} + p_{\alpha\alpha}p_{\alpha r} + p_{\alpha\beta}p_{\beta r})p_{i\alpha}$$

Since $P(S_\pi) > 0$, lemma 11 implies that p_π is identifiable. Hence

$$p_\pi = (\sum_{b \in D, b \neq \alpha,\beta} p_{\alpha b} p_{br} + p_{\alpha\alpha} p_{\alpha r} + p_{\alpha\beta} p_{\beta r}) p_{i\alpha} = \text{Known Constant}$$

Since $p_{i\alpha}, i \neq \alpha$ and $p_{\alpha b}, b \neq \alpha, \beta$ and $p_{\beta r}, r \neq \alpha, \beta$ and $p_{ab}, \quad a, b \neq \alpha, \beta$ are all estimable from the above equation we get a equation of the form

$$C_1 p_{\alpha\alpha} + C_2 p_{\alpha\beta} = K_1$$

where C_i and K_i are constants.

Also from the restriction $\sum_j p_{\alpha j} = 1$ we get a equation of the form

$$p_{\alpha\alpha} + p_{\alpha\beta} = K_2$$

These two final equations make the parameters $p_{\alpha\alpha}$ and $p_{\alpha\beta}$ identifiable.

Step 3:

Consider $p_{\beta j} \;\; j = \alpha, \beta$.

Let the β^{th} column has a element $f_{i\beta} = 1$, $i \neq \alpha, \beta$ and we choose a r such that $r \neq \alpha, \beta$.

Then $P(S_{i\beta\_r}) > 0$ which means $P(S_\pi) > 0$ where $\pi = i\beta\_r$.

Let $D = \{b : f_{\beta b} = 0 \quad \text{and} \quad f_{br} = 0\}$. Then p_π is of the form

$$p_\pi = p_{i\beta}(\sum_{b \in D} p_{\beta b} p_{br})$$

Clearly $\alpha, \beta \in D$ and hence we get

$$p_\pi = (\sum_{b \in D, b \neq \alpha,\beta} p_{\beta b} p_{br} + p_{\beta\alpha} p_{\alpha r} + p_{\beta\beta} p_{\beta r}) p_{i\beta}$$

Since $P(S_\pi) > 0$, lemma 11 implies that p_π is identifiable. Hence

$$p_\pi = (\sum_{b \in D, b \neq \alpha,\beta} p_{\beta b} p_{br} + p_{\beta\alpha} p_{\alpha r} + p_{\beta\beta} p_{\beta r}) p_{i\beta} = \text{Known Constant}$$

Since $p_{i\beta}, i \neq \alpha$ and $p_{\beta b}, b \neq \alpha, \beta$ and $p_{\alpha r}, r \neq \alpha, \beta$ and $p_{ab}, \quad a, b \neq \alpha, \beta$ are all estimable from the above equation we get a equation of the form

$$C_1 p_{\beta\alpha} + C_2 p_{\beta\beta} = K_1$$

where C_i and K_i are constants.

Also from the restriction $\sum_j p_{\beta j} = 1$ we get a equation of the form

$$p_{\beta\alpha} + p_{\beta\beta} = K_2$$

These two final equations make the parameters $p_{\beta\alpha}$ and $p_{\beta\beta}$ identifiable.

Thus all the parameters for M are identifiable. Hence for any matrix $M \in \mathbb{C}_3$, we have $M \in \mathcal{F}$. Thus $\mathbb{C}_3 \subseteq \mathcal{I}$. □

CHAPTER 5

On the Construction of optimal Filtering Mechanism

5.1. Introduction

In Chapter 2, we have introduced the general idea of filtering, where we have argued that the choice of filtering mechanism largely affects the efficiency of the parameter estimates as well as governs the size of the filtered data. In fact, in both independent and dependent samples setups we should not only be concerned with how much to retain, but we should also consider what to retain. Hence, for any data reduction problem, the issue of designing an optimal filtering mechanism is an important topic which we shall consider in this final chapter.

In Chapter 3, we have considered independent samples where the filtering mechanism is accomplished by taking a few linear combinations $y \in \mathbb{R}^m$ of the original sample points $x \in \mathbb{R}^n$ in the form $y = Ax$. Thus constructing a suitable filtering mechanism, in that case, means getting an optimal choice of the matrix A. As we have already discussed in that chapter, the existing literature of Compressive Sampling guides us how we can have such a choice of A using the Restricted Isometry Property (RIP).

Hence, in the current discussion we shall be concerned with the construction of filtering mechanism in the case of dependent samples, more specifically for Markov chains which we have introduced in Chapter 4. The idea of filtering for Markov chains was implemented through the concept of a filter matrix. We have identified three sufficient conditions for constructing a filter matrix so that the parameters in the Markov model remain estimable. This leads to three distinct classes, $\mathbb{C}_1, \mathbb{C}_2$ and $\mathbb{C}_3$, within which we shall search for our filter matrix. We have described the estimation procedure of the transition probabilities given any fixed filter matrix using EM Algorithm. However, the question of constructing such a filter matrix still remains unaddressed. We'll see in this discussion that, in order to develop a suitable method for the construction of filter matrices, we need to specify something that we call the size and information content of the observed data produced from that filter matrix. Section 2 and Section 3 formalize the trade-off between these two considerations. In section 4 we attempt to get a simple theoretical structure of the idea of the expected size of a filter matrix. Based on these developments, algorithms are devised to construct an optimal filter matrix for a given problem in section 5. These are further illustrated through a real life data application in section 6. These concepts of designing optimal filter matrices can be extended to an adaptive version where the filter matrix can "learn and adapt" to the stochastic process, as we shall discuss in section 7. Finally, in section 8, we compare two possible approaches to storing the filtered data. In section 9, we conclude the thesis with some possible directions for future developments.

5.2. Two important criteria: size and efficiency

There are primarily two considerations when we choose any filter matrix:

- the amount of storage required to store the filtered data and
- the efficiency of the estimates obtained from the filtered data

Apparently the problem is simple: the more we store, the more efficient our estimates become. So we need to define an idea of " size of storage" required by a filter matrix. We measure this using proportions of the data retained. More precisely, we use the following definition.

Definition 42. For any filter matrix F and any given input Markov chain, the size of the filter matrix is defined by

$$\frac{1}{n}\sum_{i}\sum_{j} n_{ij}$$

where n_{ij} is the number of transitions from state i to state j in the observed chain and n is the total number of transitions in the original chain.

Example 43. Consider a three state Markov chain x as

$$112312232123331121331$$

which we filter using the matrix

$$F = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}.$$

to get the filtered chain as

11␣312232␣␣␣␣311␣␣␣31.

Then the size of the filter matrix with respect to the given Markov chain is $\frac{9}{20} = 0.45$. The size of the filter matrix

$$F = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

is however 1 because in that case the filtered chain will be the same as the original chain.

However, for any fixed filter matrix, the proportion of data stored will be a random quantity varying with the input Markov chain and hence we should work with the expected value of this quantity.

Definition 44. The expected size of a filter matrix F is then defined as

$$|F| = \sum_i \sum_j \frac{E(n_{ij})}{n}.$$

The efficiency of the estimates, on the other hand, is generally measured in terms of the observed variance-covariance matrix. While we can compute this matrix numerically, we would prefer having a closed form for optimization. Since this is not always possible, we adopt a more direct approach in terms of the observed transitions in the filtered data. More specifically, suppose a single Markov chain x is filtered by two possible filter matrices F and G to create two different observed chains $\phi_F(x)$ and $\phi_G(x)$ respectively. As we know from Chapter 4, if $F \preceq G$, then $\phi_G(x)$ contains all the observed transitions present in $\phi_F(x)$. Moreover, this holds true for every possible Markov chain x. Thus, we can say that G is more *information preserving* than F is. Hence, the best possible analysis one can do with G should be as good as the best possible analysis one can do with F.

5.3. Choice of Filter Matrix: Trade-Off between Size and Efficiency

Thus, we look at the problem of an "optimal" choice of a filter matrix as a trade-off between the size and the information content of the filter matrix. Instead of attempting to solve this trade-off directly, which is not easy to do, we adopt a greedy procedure. We note that the size of a filter matrix is one quantity that can be fixed by the user well before the sampling process starts, depending on the available storage in the system. Hence, we can fix a value $\alpha \in (0, 1)$, and then restrict ourselves to the class of all filter matrices whose expected size is less than or equals to α, viz,

$$\mathcal{F}^{(\alpha)} = \{F : |F| \leq \alpha\}.$$

Then, among this class $\mathcal{F}^{(\alpha)}$ we choose the filter matrix which is most information preserving. We call this filter matrix to be optimal within $\mathcal{F}^{(\alpha)}$. The implementation of this idea, however, requires finding the expected size of a filter matrix.

5.4. Finding the expected size of a filter matrix

In principle, the expected size of a filter matrix F should depend on the original transition probability matrix P and the structure of the filter matrix itself. It turns out that finding the exact mathematical form of $|F|$ is too difficult. Hence, we shall look for a tractable approximation of the same in this section.

We first recall that for any filter matrix $F = ((f_{ij}))$ of size $m \times m$, the transitions in the filtered chain may be classified into one of the three categories:

- directly recorded ($f_{ij} = 1$)
- indirectly recorded ($f_{ij} = 0$, but the transition occurs in the filtered chain.)
- unobserved ($f_{ij} = 0$ and the transition does not appear in the filtered chain.)

Example 45. (Example 43 Continued) The filtered chain consists of some transitions which are directly recorded such as $1 \rightarrow 1$ and there may be some transitions which are indirectly recorded in the filtered chain (such as $2 \rightarrow 3$ is recorded even if $f_{23} = 0$) and some transitions like $3 \rightarrow 3$ do not appear in the filtered chain.

Then we have the following large sample approximation of the expected size of a filter matrix.

Theorem 46. *Consider a Markov chain with transition probability matrix $P_{m\times m} = ((P_{ij}))$ and total number of transitions n, which converges to a unique stationary distribution $\pi = (\pi_1, \pi_2, ...\pi_m)$ irrespective* *of the initial distribution. If we filter the Markov chain using the filter matrix* $F = ((f_{ij})) = F(P, n)$*, then the limiting value* L *of the expected size of the filter matrix is given by*

$$L = \lim_{n \to \infty} |F(P,n)| = \sum_{\substack{i \\ (i,j):f_{ij}=1}} \sum_{j} p_{ij}\pi_i + \sum_{\substack{i \\ (i,j):f_{ij}=0}} \sum_{j} p_{ij} \sum_{(\gamma,\delta)\in D_{ij}} p_{j\delta} p_{\gamma i} \pi_\gamma$$

where $D_{ij} = \{(\gamma, \delta) : f_{\gamma i} = f_{j\delta} = 1\}$ *for fixed* (i, j)*.*

This is a reasonable approximation for all practical purposes because we need to only assume the sample size n is large enough, which is also the main reason for applying the filtering mechanism at the beginning. Hence, from now onwards, we shall use this limiting form as the value of $|F|$ for a filter matrix F. Further, based on this approximated expression for $|F|$, we have the following monotonicity property of $|F|$.

THEOREM 47. *For any two filter matrices* F_1 *and* F_2*, if* $F_2 \succeq F_1$*, then* $|F_2| \geq |F_1|$*.*

The proofs of both the theorems are given in the appendix of this chapter.

5.5. Construction of filter matrices

Theorem 47 in the previous section suggests the following approach: starting from a fixed filter matrix, as we go on converting the 0 elements to 1, the expected size is either going to increase or at least remain the same. We can use this approach to provide a systematic method of finding a maximally optimal filter matrix of a pre-specified expected size α. We shall start with some filter matrix F, belonging to the class of all identifiable matrices $\overline{\mathbb{C}_*}$, and then go on converting the 0 elements to 1 as many as we can so that the expected size of the matrix is less than α. If we denote $\mathcal{F}^{(\alpha)}$ to be the set of all possible filter matrices of expected size α, then we shall consider

$$\bar{F}_\alpha = \bar{F} \cap \mathcal{F}^{(\alpha)}$$

where $\bar{F} = \{G : G \succeq F\}$. Now for every α, the set $\bar{F}_\alpha$ being finite, there must exist at least one optimal $F^*_{(\alpha)} \in \bar{F}_\alpha$, but a search for such $F^*_{(\alpha)}$ is computationally intensive procedure.

Instead, we shall adopt a greedy procedure which approximates this task of finding an optimal $F^*_{(\alpha)}$. The main idea of the greedy procedure is that starting from a fixed filter matrix F, which we shall refer to as the root matrix, we can proceed by converting the 0 elements to 1 stepwise such that the expected size of the filter matrix is increased as small as possible. More specifically, the greedy algorithm at any stage converts from 0 to 1 that entry of F which causes the least increment in $|F|$. That is, at the t^{th} iteration, we move from $f_{ij}^{(t-1)} = 0$ to $f_{ij}^{(t)} = 1$ where

$$(i, j) = \arg\min\{|F^{(t)}| - |F^{(t-1)}|\}.$$

In case of a tie, we choose any one possible (i, j). The only thing we need to assume in this greedy approach is that the size constraint α should be such that

$$\alpha \geq |F|$$

where F is the root matrix of the algorithm.

5.5.1. Determining the root structure: The greedy search algorithm, we just mentioned, works by modifying a pre-specified root matrix F. How can we determine an optimal root matrix F? We note that the initial root matrix F may belong to any one of the three classes $\mathbb{C}_1, \mathbb{C}_2$ or $\mathbb{C}_3$ because all of these classes are sufficient for identifiability. Hence, before proceeding further, let us formalize the idea of finding a root matrix into one of these classes.

DEFINITION 48. For any class of filter matrix $\mathbb{C}_i$,$i = 1, 2, 3$, a root matrix within the class $\mathbb{C}_i$ is a matrix $F = F_R(\mathbb{C}_i)$ such that

- $F \in \mathbb{C}_i$,
- if $G \succeq F$, then $G \in \mathbb{C}_i$,
- if $H \preceq F$, then $H \notin \mathbb{C}_i$.

This means for each class $\mathbb{C}_i$, we are interested in a matrix $F = ((f_{ij}))$ such that if we convert any $f_{ij} = 0$ to $f_{ij} = 1$, then F remains within $\mathbb{C}_i$ but if we convert any $f_{ij} = 1$ to $f_{ij} = 0$,the matrix F no longer remains within $\mathbb{C}_i$. The choice of such a root matrix is however not unique for a class $\mathbb{C}_i$.

EXAMPLE 49. For a three state Markov chain, both the matrices

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \text{ and } \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

are root matrices of class $\mathbb{C}_1$.

Algorithm 2 Finding root in $\mathbb{C}_1$

```
a) Fix $\mathcal{B} = \{1, 2, ..., m\}$ and $\mathcal{A} = (1, 2, ..., m\}$.
b) for each $i \in \mathcal{B}$, compute $m_i = \pi_i \min_{j \in \mathcal{A}} p_{ij}$
c) find $k$ such that $k = \arg\min_{i \in \mathcal{B}} m_i$.
d) set $f_{k\ell} = 1$ where $\ell = \arg\min_{j \in \mathcal{A}} p_{kj}$.
e) update $\mathcal{B} = \mathcal{B} - \{k\}$ and $\mathcal{A} = \mathcal{A} - \{\ell\}$.
f) Repeat steps 2 to 5 until $\mathcal{B}$ and $\mathcal{A}$ are singleton.
```

Algorithm 3 Finding root in $\mathbb{C}_2$

```
a) Fix $\mathcal{B} = \{1, 2, ..., m\}$ and $\mathcal{A} = (1, 2, ..., m\}$.
b) for each $i \in \mathcal{B}$, compute $m_i = \pi_i \min_{j \in \mathcal{A}} p_{ij}$
c) find $k$ such that $k = \arg\min_{i \in \mathcal{B}} m_i$.
d) set $f_{k\ell} = 1$ where $\ell = \arg\min_{j \in \mathcal{A}} p_{kj}$.
e) update $\mathcal{B} = \mathcal{B} - \{k\}$and $\mathcal{A} = \mathcal{A} - \{\ell\}$.
f) Repeat steps 2 to 5 until $\mathcal{B}$ and $\mathcal{A}$ contains exactly two elements.
g) For each $j \in \mathcal{A}$, set $f_{jq} = 1$ for $q \notin \mathcal{A}$.
```

The selection of a root matrix in any one of these classes is based on the greedy approach we discussed above. We start with a null matrix and go on converting $f_{ij} = 0$ to $f_{ij} = 1$ so that $|F|$ is increased as small as possible while the structure of the class is maintained. For example, while selecting a root in $\mathbb{C}_1$, if we convert $f_{ij} = 0$ to $f_{ij} = 1$, then at the subsequent stages we shall discard the i^{th} row and the j^{th} column from further consideration. Thus we need to have three separate algorithms for finding a root matrix in each of these three classes.

With the application of Algorithms 2, 3 and 4, we shall select one possible root matrix from each of the three classes. Among these three root matrices, we shall select the one as our final root matrix for which

Algorithm 4 Finding root in $\mathbb{C}_3$

```
a) Fix $\mathcal{B} = \{1, 2, ..., m\}$ and $\mathcal{A} = (1, 2, ..., m\}$.
b) for each $i \in \mathcal{B}$, compute $m_i = \pi_i \min_{j \in \mathcal{A}} p_{ij}$
c) find $k$ such that $k = \arg\min_{i \in \mathcal{B}} m_i$.
d) set $f_{k\ell} = 1$ where $\ell = \arg\min_{j \in \mathcal{A}} p_{kj}$.
e) update $\mathcal{B} = \mathcal{B} - \{k\}$and $\mathcal{A} = \mathcal{A} - \{\ell\}$.
f) Repeat steps 2 to 5 until $\mathcal{B}$ and $\mathcal{A}$ contains exactly two elements.
g) For each $j \in \mathcal{B}$, set $f_{qj} = 1$ for $q \notin \mathcal{B}$.
```

$|F|$ is minimum. That is, we select our root matrix as

$$F = \arg\min |F_R(\mathbb{C}_i)|.$$

It turns out that the restriction on the size constraint α becomes

$$\alpha \geq \min_i |F_R(\mathbb{C}_i)|.$$

5.5.2. Modifying the root matrix: Once we have the final root matrix F in our hand, we can apply the same greedy approach to modify it stepwise by converting $f_{ij} = 0$ to $f_{ij} = 1$ which causes the least possible increment in $|F|$. We shall continue this process till the size of the matrix exceeds α. It turns out that, even this greedy searching procedure can be computationally slow at times. Hence, in order to make the algorithm even faster, we can adopt another layer of approximation, although that is optional.

We note that the limiting form of the expected the size of a filter matrix is given by

$$L = \sum_{i} \sum_{\substack{j \\ (i,j):f_{ij}=1}} p_{ij}\pi_i + \sum_{i} \sum_{\substack{j \\ (i,j):f_{ij}=0}} p_{ij} \sum_{(\gamma,\delta)\in D_{ij}} p_{j\delta}p_{\gamma i}\pi_\gamma = |F|_{(1)} + |F|_{(2)}, \text{say.}$$

From the above expression, we note that at any stage t, the greedy search algorithm is relatively faster and computationally easy if we take $L = |F|_{(1)}$. This is because while comparing different possible options of filter matrices, calculating $|F|_{(2)}$ is computationally intensive because we need to track all the indirect transitions as well. However, $L = |F|_{(1)}$ is an under-estimation of the size of the filter matrix and this approximation will be very crude when $|F|_{(2)}$ assumes a significantly large value. Now $|F|_{(2)}$ assumes a significantly large value when there are many indirectly recorded transitions in the observed chain, which again increases with the number of directly recorded transitions in the chain. Hence, in order to maintain the scalability of the search algorithm, it is recommended that in the process of modifying the root matrix F, we should use the approximation $L = |F|_{(1)}$ during the initial stages and after a certain stage t, we use the original expression $L = |F|_{(1)} + |F|_{(2)}$. This is because, during the initial stages of the algorithm, the number of direct transitions in the chain will be less (most of $f_{ij} = 0$) and as such $|F|_{(2)}$ is insignificant whereas, during the latter stages of the algorithm we have converted many entries in the filter matrix from 0 to 1 which makes $|F|_{(2)}$ significantly larger. While implementing the algorithm, the user can control this optional approximation to make the algorithm faster with a stage parameter t, which determines the stage until which we shall use the faster approximation in the algorithm. Setting $t = 0$, means that we are not using the second layer of approximation at all, but then the algorithm can be potentially slow. On the other hand, setting t to be a very large value yields a potentially fast algorithm but with this additional layer of approximation which can give crude approximation. Algorithm 5 thus provides us a systematic search procedure to find the optimal filter matrix starting from a root form F and given a user specified size constraint α.

5.6. Practical application

We shall illustrate the applicability of the above procedures with a real life data set. Androsensor is one of the free softwares available in Google Play Store developed as an all-in-one diagnostic tool for smartphones. This software can capture many essential information like accelerometer readings, gyroscope readings, ambient magnetic field values, device orientation, proximity sensor readings from the user's smartphone on the go.

We shall work with a data set which contains 20031 readings on linear acceleration (along X-axis) of the author's smartphone during 3 hours of a specified day. A glimpse of the data is shown in Table 5.6.1.

In the data pre-processing step we create a discrete variable with 4 levels by splitting the data using the quartiles $Q_1 = -0.0042650$, $Q_2 = .0000800$ and $Q_3 = .004535$. A glimpse of the discretized data is provided in Table 5.6.2.

Algorithm 5 Finding optimal filter matrix

```
Input: size $\alpha$, stage parameter $t$, transition probability matrix $P$ and
    stationary distribution $\pi$.
  • Call Algorithms 2, 3 and 4 to find $F_R(\mathbb{C}_i), i = 1, 2, 3$ and set
        $F = \arg\min |F_R(\mathbb{C}_i)|.$
  • Set $w = 0$
  • while $w < t$
      a) for the $i^{th}$row, fix the active set as $\mathcal{A}_i = \{j : f_{ij} = 0\}$
      b) for each $i$, compute $m_i = \pi_i \min_{j \in \mathcal{A}_i} p_{ij}$
      c) find $s$ such that $s = \arg\min_i m_i$.
      d) set $f_{s\ell} = 1$ where $\ell = \arg\min_{j \in \mathcal{A}_s} p_{sj}$.
      e) update $\mathcal{A}_s = \mathcal{A}_s - \{\ell\}$and $w = w + 1$.
      f) if $|F| > \alpha$, break.
  • while $w \geq t$
      a) for the $i^{th}$row, fix the active set as $\mathcal{A}_i = \{j : f_{ij} = 0\}$
      b) for each $i$, compute $M_i = \min_{j \in \mathcal{A}_i} |F|_{(i,j)}$ where $|F|_{(i,j)}$ is the
         size of $F$ after making $f_{ij} = 1$.
      c) find $s$ such that $s = \arg\min_i M_i$.
      d) set $f_{s\ell} = 1$ where $\ell = \arg\min_{j \in \mathcal{A}_s} |F|_{(s,j)}$.
      e) update $\mathcal{A}_s = \mathcal{A}_s - \{\ell\}$and $w = w + 1$.
      f) if $|F| > \alpha$, break.
    Output: $\hat{F} \in \bar{F}_\alpha$ which is our greedy estimate for $F^*_{(\alpha)}$.
```

Then we can assume that the data arises from a one-stage discrete Markov process with four possible states. If we store all the observations in original Markov chain, then the maximum likelihood estimate of the transition probability matrix is

$$\hat{P}_{com} = \begin{bmatrix} 0.2987817 & 0.1887358 & 0.1843419 & 0.3281406 \\ 0.1762475 & 0.3133733 & 0.3277445 & 0.1826347 \\ 0.1898102 & 0.3204795 & 0.3226773 & 0.1670330 \\ 0.3350639 & 0.1779153 & 0.1647364 & 0.3222843 \end{bmatrix}.$$

TABLE 5.6.1. Glimpse of the data

Sl. No.	Linear Acceleration	Time
1	-0.27789	18:08:25:520
2	-0.37871	18:08:26:020
3	0.33692	18:08:26:520
4	1.37086	18:08:27:020
5	-0.17113	18:08:27:520
6	1.48150	18:08:28:019
⋮	⋮	⋮
Sl. No.	Linear Acceleration	Time
⋮	⋮	⋮
1	0.78671	20:55:18:020
2	-0.32992	20:55:18:520
3	-2.32029	20:55:19:020
4	-3.01193	20:55:19:520
5	0.17800	20:55:20:020
6	-0.07406	20:55:20:520

A part of the data showing the linear acceleration (along $X-$axis) along with the time stamps.

TABLE 5.6.2. A glimpse of the discretized data

1 1 4 4 1 4 4 4 1 1 1 1 1 1 4 4 1 1 1 1 4 1 4 4 4 1 1 1 4 4 4 4 4 1 4 4 4
1 1 4 1 1 1 1 1 1 1 4 4 4 1 4 1 1 4 4 4 1 1 1 4 4 4 4 1 1 4 4 4 4 4 4 4 4
1 4 4 1 1 4 1 1 1 4 4 1 4 1 1 4 1 4 1 1 1 4 1 4 4 1
⋮
2 1 3 1 2 2 2 3 4 4 2 4 2 3 4 1 2 3 4 2 2 2 4 1 1 2 4 3 3 4 3 1 3 2 3 1 4
1 4 4 1 4 1 4 4 1 1 4 1 4 4 4 4 4 4 4 4 1 4 4 4 4 1 1 1 1 1 4 4 1 4 1 4 4
4 1 4 4 1 4 4 1 1 4 4 1 1 4 1 4 1 1 4 1 1 4 1 1 1 4 1

However, we shall construct an optimal filter matrix to generate the filtered data. All the above algorithms of construction of a filter matrix depends on a transition probability matrix and hence we get an estimate of the same based on the initial 5000 values of the Markov chain. as

$$\hat{P}_{init} = \begin{bmatrix} 0.4466280 & 0.03439035 & 0.03126396 & 0.4877177 \\ 0.2862903 & 0.20161290 & 0.22177419 & 0.2903226 \\ 0.2910798 & 0.24882629 & 0.18309859 & 0.2769953 \\ 0.4810787 & 0.02957808 & 0.02131361 & 0.4680296 \end{bmatrix}.$$

Based on this estimate of the transition probability matrix and setting the value of $\alpha = 0.2$, we run the algorithm for finding the root matrix. The roots in each class comes out to be

$$F_R(\mathbb{C}_1) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \text{ with size } 0.2178$$

$$F_R(\mathbb{C}_2) = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix} \text{ with size } 0.0706.$$

$$F_R(\mathbb{C}_3) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \text{ with size } 0.0715$$

Hence the optimal root matrix turns out to be

$$F = F_R(\mathbb{C}_2) = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$

From this root matrix we start modifying with the stage parameter $t = 3$ to obtain the optimal filter matrix to be

$$\hat{F} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$

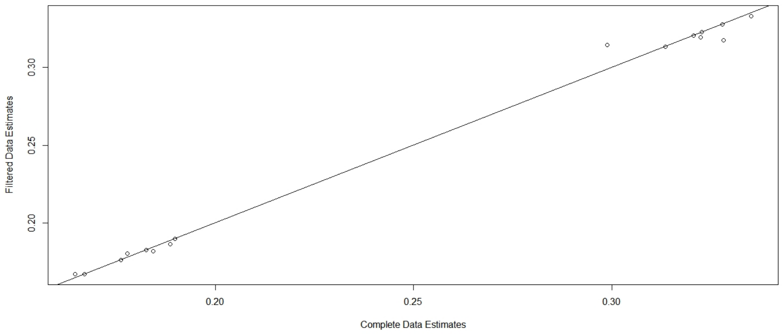
We use this matrix $\hat{F}$ to filter the remaining observations, based on which the estimate of the transition probability matrix comes out to be

$$\hat{P} = \begin{bmatrix} 0.3142931 & 0.1862983 & 0.1819612 & 0.3174475 \\ 0.1762475 & 0.3133733 & 0.3277445 & 0.1826347 \\ 0.1898102 & 0.3204795 & 0.3226773 & 0.1670330 \\ 0.3330495 & 0.1804196 & 0.1670551 & 0.3194758 \end{bmatrix}.$$

5.7. Adaptive filtering mechanism

The idea of adaptive filtering mechanism was introduced in Chapter 2, where we argued that the construction of a filtering mechanism is generally entangled with the estimation of parameter of the underlying population. This is true even for the construction of optimal

FIGURE 5.6.1. Estimates based on complete data as compared to filtered data



Estimates of transition probabilities based on complete data and filtered data are. A $y = x$ line is added for the reference which shows the agreement of the probabilities.

filter matrices as we have already discussed in the previous sections. However, in our real life data application, we have overcome this issue by constructing an initial estimate $\hat{P}_{init}$ of the transition probability matrix based on the first 5000 values of the chain and thereby use this estimate to construct the optimal filter matrix to filter the remaining part of the chain.

In practice, if we have a steady flow of observations from a Markov model, we can extend the above idea. In such cases, we can completely observe k initial observations of the chain to get an estimate of the transition probability matrix $\hat{P}$. Based on this estimate we can construct an optimal filter matrix F_1 which can be used to filter the next observations. This filtration process goes on till we intervene after a certain user specified stage (say k^*) and then construct another estimate of the transition probability matrix $\hat{Q}$. If we find that

$$\max |\hat{P}_{ij} - \hat{Q}_{ij}| < \delta,$$

for some specified δ, we shall continue with the filter matrix F_1. Otherwise we conclude that there has been a change in the underlying Markov process and we shall use the above algorithms to create the next optimal filter matrix F_2 based on $\hat{Q}$ and so on. In this way we can make the process of construction of filter matrix adaptive to the stream of observations from the Markov model.

5.8. Storing the Filtered Data

There are two ways in which we can store the observed chain. Both the methods depend on the choice of filter matrix to determine which transitions we record. At the outset we clarify that filter matrix determines the storage in terms of transitions and the missingness occurs in the chain in terms of the states. For example, suppose $f_{25} = 0, f_{12} = 1, f_{53} = 0$ and $f_{32} = 1$ in the filter matrix and some portion of the observed chain is 12532, then the filtering mechanism makes the state 5 missing. However we can express this filtered chain in two ways:

a) In the first method, the filtered chain will be in the same order of appearance as the observed chain with the missing states indicated by a symbol. Thus for the previous example, the observed chain we look like

$$12\_\_32$$

b) Alternatively we can discard the missing states and store only the observed states along with the time-stamps. For the previous example, we shall have

Time index	1	2	4	5
Observation	1	2	3	2

One possible question of interest is that which method will be more efficient in terms of storage? Suppose the original Markov chain is of length n and in the filtered chain we store only m states. Let ϵ be the storage required for the missingness symbol. Then it is reasonable to assume that ϵ is the storage required for one unit data because the missingness symbol is determined so that it requires the minimal storage. We shall call this ϵ as the storage complexity of the string. Then the proportion of storage required in the filtered chain as compared to the observed chain is

$$p_1 = \frac{m + (n-m)\epsilon}{n} = \epsilon + (1-\epsilon)\frac{m}{n}.$$

On the other hand, suppose δ be the additional storage required for storing the time-stamp. Then for the second method, the proportion of storage required is

$$p_2 = \frac{m}{n}(1+\delta).$$

Thus the first method will be preferred to the second method if and only if

$$\frac{\epsilon}{\delta} < \frac{{}^{m}\!/_{n}}{1 - {}^{m}\!/_{n}}.$$

However in the above expression m is practically not known before the filtering process starts. Hence we replace $\frac{m}{n}$ in the above expression by its expected value. Thus if $|F|$ is the size of a filtering matrix, then we shall choose the first method storing the data if

$$\frac{\epsilon}{\delta} < \frac{|F|}{1 - |F|}$$

and choose the second method otherwise.

5.9. Concluding Remarks

In this thesis, we have discussed the idea of data compression by deliberately introduced missingness. We have shown how a standard, yet simple tool like EM Algorithm can be employed to get the estimates of the parameters in such problems. This work explores a completely new area of Statistics which we feel should be more relevant in the future days, specially when Statistics as a discipline has been considering potentially large datasets as fields of application. We have considered the idea of filtering mechanism and estimation based on the filtered data for both the independent and dependent data setup. In fact, we have shown how the above-mentioned filtering mechanism can be made adaptive to the data generation process, so that we can use it in practice for most real life applications where we have a continuous stream of observations and a crisis of data storage. Since it is a pioneering work in this direction, we have considered simple data setup like Markov process yet, which in no way demeans the scope of application of the methods. However, there is further scope of generalizing the concept to more complex data models. Moreover, we can further improve the filtering mechanism so that the estimation process based on the non-ignorable missing data mechanism becomes easier.

5.10. Appendix

5.10.1. Proof of Theorem 46. First we restrict ourselves to the directly recorded transitions only. Then we want to compute

$$\underset{(i,j):f_{ij}=1}{\sum_i \sum_j} \frac{E(n_{ij})}{n}.$$

For a Markov chain x, let us define

$$U_k = \begin{cases} 1 & \text{if } x_k = i, x_{k+1} = j \\ 0 & \text{otherwise} \end{cases}, k = 0, 1, 2, ...$$

where x_0 is the initial state of the Markov chain. Then $n_{ij} = \sum U_k$ and $E(n_{ij}) = \sum P(U_k = 1)$. Now

$$P(U_k = 1) = P(x_k = i, x_{k+1} = j)$$

$$= P(x_{k+1} = j | x_k = i) P(x_k = i)$$

$$= p_{ij} P(x_k = i) = p_{ij} P^k_{0,i}.$$

where $P^k_{1,i}$ is the $(1, i)^{th}$ element of P^k which indicates the probability of visiting the state i in k steps. Thus we get

$$E(n_{ij}) = \sum_{k=0}^{n-1} p_{ij} P^k_{1,i} = p_{ij} \sum_{k=0}^{n-1} P^k_{1,i} = p_{ij} \Big(\sum_{k=0}^{n-1} P^k \Big)_{1,i}$$

and as such

$$\frac{E(n_{ij})}{n} = p_{ij}\Big(\frac{1}{n}\sum_{k=0}^{n-1} P^k\Big)_{1,i}.$$

Now assuming the Markov chain converges to stationary distribution π, we have some m, such that for $k \geq m$, $P^k \approx 1\pi^T$. This implies for large n,

$$\frac{1}{n}\sum_{k=0}^{n-1} P^k = \Pi$$

where $\Pi = (\pi, \pi, ..., \pi)$ is a matrix with all columns π. Thus we get that for large n,

$$\lim_{n\to\infty} \frac{E(n_{ij})}{n} = p_{ij}\pi_i$$

where $\pi = (\pi_1, \pi_2, ..., \pi_k)$. Finally we have

$$\lim_{n\to\infty} \sum_{\substack{i \\ (i,j):f_{ij}=1}} \sum_{j} \frac{E(n_{ij})}{n} = \sum_{\substack{i \\ (i,j):f_{ij}=1}} \sum_{j} p_{ij}\pi_i.$$

Now let us consider the case of indirect transitions, that is, (i, j) such that $f_{ij} = 0$. As before for $k = 0, 1, 2, ...$, let

$$U_k = \begin{cases} 1 & \text{if } x_k = i, x_{k+1} = j, x_{k-1} = \gamma, x_{k+2} = \delta \\ 0 & \text{otherwise} \end{cases},$$

where γ and δ are such that $f_{\gamma i} = 1$ and $f_{j\delta} = 1$ and x_0 is the initial state of the Markov chain Then $n_{ij} = \sum U_k$ and $E(n_{ij}) = \sum P(U_k = 1)$. Now for fixed (i, j)let us indicate $D_{ij} = \{(\gamma, \delta) : f_{\gamma i} = f_{j\delta} = 1\}$ and hence

$$P(U_k = 1) = \sum_{(\gamma,\delta)\in D_{ij}} P(x_k = i, x_{k+1} = j, x_{k-1} = \gamma, x_{k+2} = \delta)$$

$$= \sum_{(\gamma,\delta)\in D_{ij}} p_{ij}p_{j\delta}p_{\gamma i}P_{1\alpha}^{(k-1)}$$

$$= p_{ij} \sum_{(\gamma,\delta)\in D_{ij}} p_{j\delta}p_{\gamma i}P_{1\gamma}^{(k-1)}.$$

Thus we get

$$\lim_{n\to\infty} \frac{E(n_{ij})}{n} = \lim_{n\to\infty} p_{ij} \sum_{(\gamma,\delta)\in D_{ij}} p_{j\delta}p_{\gamma i}\Big(\frac{1}{n}\sum_{k=0}^{n-1} P^k\Big)_{1,i}$$

$$= p_{ij} \sum_{(\gamma,\delta)\in D_{ij}} p_{j\delta}p_{\gamma i}\pi_\gamma.$$

Then we have

$$L = \lim_{n\to\infty} |F(P,n)| = \sum_{\substack{i \\ (i,j):f_{ij}=1}}\sum_{j} p_{ij}\pi_i + \sum_{\substack{i \\ (i,j):f_{ij}=0}}\sum_{j} p_{ij} \sum_{(\gamma,\delta)\in D_{ij}} p_{j\delta}p_{\gamma i}\pi_\gamma$$

5.10.2. Proof of Theorem 47.

Proof. Consider two filter matrices $F_1 = ((f_{pq}^{(1)}))$ and $F_2 = ((f_{pq}^{(2)}))$ such that

$$f_{pq}^{(1)} = f_{pq}^{(2)}, p \neq i, q \neq j$$

and

$$f_{ij}^{(1)} = 0 \text{ and } f_{ij}^{(2)} = 1.$$

This means that F_1 and F_2 are identical at all positions except the $(i,j)^{th}$ position and $F_2 \succeq F_1$. It is enough to show $|F_2| \geq |F_1|$. From the expression of the expected size of a filter matrix we note that

$$|F_2| - |F_1| = p_{ij}\pi_i - p_{ij} \sum_{(\gamma,\delta)\in D_{ij}} p_{j\delta}p_{\gamma i}\pi_\gamma$$

$$= p_{ij}(\pi_i - \sum_{(\gamma,\delta)\in D_{ij}} p_{j\delta}p_{\gamma i}\pi_\gamma).$$

The result holds trivially if for any (i, j), $D_{ij} = \emptyset$. For non-empty sets D_{ij} we note the following. π being the stationary distribution of the Markov chain we have

$$\pi P = \pi$$

which implies

$$\pi_i = \sum_\gamma p_{\gamma i}\pi_\gamma$$

$$\Rightarrow \pi_i \geq \sum_{(\gamma,\delta)\in D_{ij}} p_{j\delta}p_{\gamma i}\pi_\gamma$$

which in turn implies

$$|F_2| \geq |F_1|.$$

□

Appendix

R codes for Chapter 1

```r
wave = rep(c(rep(1,10),rep(0,20)),30)
noise = rnorm(length(wave),sd=0.03)
comp = wave+noise
w = 37
nw = floor(length(wave)/w)
n = w*nw
thresh = 0.5
trig =function() {
    ON = 0; OFF1 = 1; OFF2 = 2
    state = ON
    bag = c()
    showTime = 0
    for(i in 1:length(comp)) {
        inp = comp[i]
        if(state==ON) {
            if(showTime<=0) {
                showTime = 0
                if(inp >= thresh)
                    state = OFF1
                else
                    state = OFF2
            }
            else {
                showTime = showTime - 1
            }
        }
        else if(state==OFF1) {
            if(inp < thresh) state = OFF2
        }
        else {
            if(inp > thresh) {
                state = ON
                showTime = w
                bag = c(bag,i)
            }
        }
    }
    bag
}
##svg('oscil%draw.svg')
### Full plot 1
plot(comp,ty='l',ylim=c(-0.5,1.5),xlab="Time(t)",ylab="Voltage(V)")
for(i in 1:nw) {rect(w*(i-1),-0.2,w*i,1.2)}

### Collapsed plot 1
plot(comp[1:w],ty='l',xlab="Time(t)",ylab="Voltage(V)")
for(i in 2:nw) {rng = w*(i-1)+ (1:w); lines(comp[rng])}

trigStarts = trig()

### Full plot 2
plot(comp,ty='l',ylim=c(-0.5,1.5),xlab="Time(t)",ylab="Voltage(V)")
for(i in 1:length(trigStarts))
{rect(trigStarts[i],-0.2,trigStarts[i]+(w-1),1.2)}
abline(h=comp[trigStarts[1]],col="grey")
##par(new=F)
```

```
for(i in 2:length(trigStarts))
{lines(x=(trigStarts[i-1]+(w-1)):trigStarts[i],
y=comp[(trigStarts[i-1]+(w-1)):trigStarts[i]],type="l",col="red")}

### Collapsed plot 2
plot(0,xlim=c(1,w),ylim=c(-0.5,1.5),ty=’n’,
xlab="Time(t)",ylab="Voltage(V)")
for(ts in trigStarts) {
    lines(1:w,comp[ts:(ts+(w-1))])
}
dev.off()
```

R codes for Chapter 2

```
#-----setting up the parameters
start=0
phi=0.5
sigma=1
n=100
##c=0.1
c_seq=seq(0.01,5,by=0.1)
phi_c1=NULL
phi_c2=NULL
miss_c=NULL
k=1
for(c in c_seq)
{
phi_simul=NULL
set.seed(200)

for(j in 1:100)
{
```

```
#-----generation of data
x=NULL
x[1]=start
for (i in 2:(n+1))
{
        x[i]=phi*x[i-1]+rnorm(1,mean=0,sd=sigma)
}
#------filteration of data
y=NULL
y[1]=x[1]
for (i in 2:(n+1))
{
        if(abs(x[i]-x[i-1])>c | is.na(y[i-1])==T)
                y[i]=x[i]
        else
                y[i]=NA
}
miss_c[k]=sum(is.na(y))
#------Estimation starts---

phi0=0
phi1=0.2

while(abs(phi1-phi0)>0.000001)
{
        phi0=phi1
        #----calculation of E1---
        z=y[-(n+1)]
        part11=sum(z^2,na.rm=T)
        part12=0
        for (i in 1:n)
```

```
{
        if (is.na(y[i]))
        {
        alpha=(y[i-1]-c-(phi0*y[i-1]))/sigma
        beta=(y[i-1]+c-(phi0*y[i-1]))/sigma
        Z=pnorm(beta)-pnorm(alpha)
        temp1=(phi0*y[i-1])^2
        temp2=(sigma^2)*((alpha*dnorm(alpha))-(beta*dnorm(beta)))/Z
        temp3=(dnorm(alpha)-dnorm(beta))/Z
        temp=(sigma^2)+temp1+temp2+(2*sqrt(temp1)*sigma*temp3)
        part12=part12+temp
        }
}
E1=part11+part12

#-----calculation of E2----
part1=0
for(i in 2:(n+1))
{
        if(is.na(y[i])==F & is.na(y[i-1])==F)
        part1=part1+(y[i]*y[i-1])
}

part2=0
for(i in 2:(n+1))
{
        if(is.na(y[i]))
        {
                alpha=(y[i-1]-c-(phi0*y[i-1]))/sigma
                beta=(y[i-1]+c-(phi0*y[i-1]))/sigma
                Z=pnorm(beta)-pnorm(alpha)
                temp3=(dnorm(alpha)-dnorm(beta))/Z
```

```
                        temp=(phi0*y[i-1])+(sigma*temp3)
                        part2=part2+(temp*y[i-1])

                }
        }

        part3=0
        for(i in 2:(n+1))
        {
                if(is.na(y[i-1]))
                {
                        alpha=(y[i-2]-c-(phi0*y[i-2]))/sigma
                        beta=(y[i-2]+c-(phi0*y[i-2]))/sigma
                        Z=pnorm(beta)-pnorm(alpha)
                        temp3=(dnorm(alpha)-dnorm(beta))/Z
                        temp=(phi0*y[i-2])+(sigma*temp3)
                        part3=part3+(temp*y[i])
                }


        }
        E2=part1+part2+part3
        phi1=E2/E1
}

phi_simul[j]=phi1
}
phi_c1[k]=mean(phi_simul)
phi_c2[k]=var(phi_simul)
k=k+1
}
##par(mar=rep(0.1,4))
```

```
##par(mfrow=c(1,2))
plot(c_seq,phi_c1,type="b",xlab="c",ylab=expression(hat(phi)))
abline(h=phi,col="grey")
plot(c_seq,phi_c2,type="b",xlab="c",ylab=expression(hat(Var(phi))))

plot(c_seq,(miss_c/n),type="b", xlab="c",
ylab="Proportion of discarded observations")
```

```
par(mfrow=c(2,2))
theta_simul=NULL
for(i in 1:1000)
{
theta=5
n=100
x=rexp(n, 1/theta)

#------------filtering process----
discretize=function(data,breaks,k=4)
{
        return(cut(data,br=c(min(x),breaks,max(data)),
        labels=F,right=FALSE,include.lowest=TRUE))
}
mybreaks=c(1,1.5,2,5,10,15)
y=discretize(x,breaks=mybreaks)
freq=table(y)
k=length(breaks)+1
if (length(freq)<k)
        {
                nam=as.numeric(names(freq))
                index=setdiff(1:k,nam)
        }
```

```r
#---------EM starts------
breaks=mybreaks
b=breaks
a=c(0,breaks[-length(breaks)])
theta1=0
theta2=1
while(abs(theta2-theta1)>0.0000000001)
        {
                theta1=theta2
                temp=(b*exp(-b/theta1)-a*exp(-a/theta1))/(exp(-b/theta1)-exp(-a/theta1))
                temp=c(temp,breaks[length(breaks)])
                xcap=theta1+temp
                if(length(xcap)>length(freq))
                theta2=sum(freq*xcap[-index])/n
                else
                theta2=sum(freq*xcap)/n
        }
theta_simul[i]=theta2
}
title=mybreaks
hist(theta_simul,main=paste(title,collapse=","),xlab=expression(hat(theta)))
```

R codes for Chapter 3

```r
require(mvtnorm)
require(MASS)
require(R1magic)
#-------------Defining the constants------------
n=100
m=80
k=80
truek=4
```

```
#--------------Defining true mew ------------
sig=c(seq(0.1,1,0.2))
siglength=length(sig)

mew=c(rep(5,truek),rep(0,n-truek))
Usual_Resid= numeric(10)
Naive_Resid=numeric(10)
New_Resid=numeric(10)
U_mean=numeric(siglength)
N_mean=numeric(siglength)
New_mean=numeric(siglength)
U_sd=numeric(siglength)
N_sd=numeric(siglength)
New_sd=numeric(siglength)

for (sigcount  in 1:siglength)
{

 for( simul in 1:10)
 {

  #----Constructing the sensing matrix and other related matrices-----
  temp=rnorm((m*n),mean=0,sd=sqrt(1/m))
  phi=matrix(temp,nrow=m)
  phiinv=solve(phi%*%t(phi))
  coeff=t(phi)%*%phiinv

  #------Applying Usual Compressive Sensing Algorithm------

 dist=NULL   #---a vector string the L2 norm of distances at each simulation---

  for (i in 1:5)
```

```
    {
      x=rmvnorm(1,mean=mew,sigma=sig[sigcount]*diag(n))
      y=phi %*% t(x)
      T <- diag(n) ;# Do identity transform
      p <- matrix(0, n, 1) ;# initial guess
      # R1magic Convex Minimization !
      # (unoptimized penalty parameter)
      ll <- solveL1(phi, y, T, p)
      x1 <- ll$estimate ;# Returns nlm obje
      dist[i]=(x-x1)%*%t(x-x1)
    }

    Usual_Resid[simul]=mean(dist)     #--------Distance of old algorithm



    #-----Applying New Proposed Algorithm------


    x=rmvnorm(1,mean=mew,sigma=sig[sigcount]*diag(n))
    y=phi %*% t(x)
    K=coeff%*%y
    Beta=coeff%*%phi
    Varmat=diag(n)-Beta
    varx=diag(Varmat)
    pen=ginv(Beta)



    #-----Creating function to construct the mean
#------vector from current estimates of mu------
    mut=function(x)
    {
      temp=x+K-Beta%*%x
      return(temp)
    }

    #------Function to compute the maximised value of Q-----
    Q=function(index,mu)
    {
      tempindex=setdiff(1:n,index)
      current_mean=mut(mu)
      temp=sum((current_mean[index])^2)+
        sum(varx[index])+sum(varx[tempindex])
      return(temp)

    }


    #----------Applying Modified Algorithm-------

    #--------finding the subspace-------

    #---------------Unrestricted EM-------

    mu1=rep(0.00001,n)
    mu2=mu1+10
    temp1=mu1
    while(any(abs(mu2-mu1)>0.0000000001))
    {
      mu1=temp1
      mu2=mut(mu1)
      temp1=mu2

    }

    #--------Constructing variance of muhat-------
```

```
tempvar= pen%*%t(phi)%*%phiinv%*%(phi)%*%t(pen)



ind=which(abs(mu2/sqrt(sig[sigcount]*diag(tempvar)))>2.575829)
if(length(ind)>m)
{
  tempind=sort(abs(mu2/sqrt(sig[sigcount]*diag(tempvar))),
    decreasing=TRUE,index.return=TRUE)
  ind=tempind$ix[1:m]
}



#----------------------Now Final EM over Restricted Subspace------


mu1=rep(2,n)
temp1=mu1
mu2=mu1+10

while(any(abs(mu2-mu1)>0.0000000001))
{
  mu1=temp1
  temp=mut(mu1)
  mu2=rep(0,n)
  mu2[ind]=temp[ind]
  temp1=mu2
}

restricted_est=mu2
```

```
    New_Resid[simul]= t(mew-restricted_est)%*%(mew-restricted_est)

                          #----distance of new algorithm---




  }





  U_mean[sigcount]=mean(Usual_Resid)



  New_mean[sigcount]=mean(New_Resid)

  U_sd[sigcount]=sd(Usual_Resid)/10



  New_sd[sigcount]=sd(New_Resid)/10

}



plot(sig,U_mean,type="b",ylab="",ylim=c(0,90),

yaxs="i",lty=1,lwd=2,xlab=expression(sigma),

main="Average residuals",sub="n=100,m=80")



lines(sig,New_mean,type="b",lwd=2,lty=2)



segments(x0=sig,y0=New_mean-New_sd,y1=New_mean+New_sd)

segments(x0=sig,y0=U_mean-U_sd,y1=U_mean+U_sd)



legend(x=0.4,y= 90,lty=c(1,2),c("Conventional","ESREM"),cex=1,bty="n",lwd=2)
```

R codes for Chapter 4

```
#----finding the optimal filter matrix-----



#---sample P  and F matrix
```

```r
##P=matrix(c(0.1,0.2,0.7,0.5,0.2,0.3,0.2,0.4,0.4),byrow=T,nrow=3)
##F=matrix(c(1,1,0,0,0,1,0,1,0),byrow=T,nrow=3)

#----function finding the stationary distribution---

stationary=function(P)
{
        temp=as.numeric(eigen(t(P))$vec[,1])
        return(temp/sum(temp))
}


#-----function finding the size of a filter matrix

size=function(F,P,both=TRUE)
{
        pi=stationary(P)
        temp=P*pi
        F1=sum(temp[F==1])

        if(both)
        {

                for (i in 1:nrow(F))
                {
                        for(j in 1:ncol(F))
                        {
                                if (F[i,j]==0)
                                {
                                Da=which(F[,i]==1)
                                Db=which(F[j,]==1)
                                D=expand.grid(Da,Db)
                                temp=0
                                if(nrow(D)>0)
                                for (k in 1:nrow(D))
                                {
                                alpha=D[k,1]; beta=D[k,2]
                                temp=temp+P[j,beta]*P[alpha,i]*pi[alpha]
                                }
                                F1=F1+(temp*P[i,j])
                                }
                        }
                }
        }
return(F1)
}



#----root in C1-----
root1=function(P)
{
pi=stationary(P)
K=nrow(P)
F=matrix(0,nrow=K,ncol=K)
B=1:K; A=1:K
m=NULL
while(length(B)>1)
{
for (i in B)
m[i]=pi[i]*min(P[i,A])
k=B[which.min(m[B])]
l=A[which.min(P[k,A])]
F[k,l]=1
```

```
B=setdiff(B,c(k)); A=setdiff(A,c(l))
}
return(F)
}

#-----root in C2 and C3---

root23=function(P)
{
pi=stationary(P)
K=nrow(P)
F1=F2=matrix(0,nrow=K,ncol=K)
B1=B2=1:K; A1=A2=1:K
m1=m2=NULL
while(length(B1)>2)
{
for (i in B1)
m1[i]=pi[i]*min(P[i,A1])
k=B1[which.min(m1[B1])]
l=A1[which.min(P[k,A1])]
F1[k,l]=1
F2[k,l]=1
B1=setdiff(B1,c(k)); A1=setdiff(A1,c(l))
B2=B1; A2=A1
}
set1=setdiff((1:K),B1)
set2=setdiff((1:K),A2)
for(j in A1)
F1[j,set1]=1
for(j in B2)
F2[set2,j]=1
```

```
return(list(F1,F2))
}

#----compare the sizes of the root
compare=function(P)
{
r1=root1(P)
print("r1 done")
obj=root23(P)
r2=obj[[1]]
print("r2 done")
r3=obj[[2]]
r=list(r1,r2,r3)
size1=size(r1,P,both=FALSE)
size2=size(r2,P,both=FALSE)
size3=size(r3,P,both=FALSE)
index=which.min(c(size1,size2,size3))
return(r[[index]])
}

#-----construct the filter matrix---

optfilter=function(level,root,stage=3,P)
{
        pi=stationary(P)
        A=list()
        for(i in 1:nrow(P))
                A[[i]]=which(root[i,]==0)
        iter=1
        M=NULL
        F1=F2=root
        tempsize=size(F1,P)
        while(iter<=stage)
        {
                F1=F2
                for (i in 1:nrow(P))
                M[i]=pi[i]*min(P[i,A[[i]]])
                k=which.min(M)
                l=A[[k]][which.min(P[k,A[[k]]])]
                F1[k,l]=1
                A[[k]]=setdiff(A[[k]],c(l))
                iter=iter+1; print(iter)
                tempsize=size(F1,P,both=FALSE)
                if(tempsize>=level)
                break
                F2=F1
        }

        while(iter>stage)
        {       change=matrix(1000,nrow=nrow(P),ncol=ncol(P))
                F1=F2
                for (i in 1:nrow(F1))
                {
                        for(j in 1:ncol(F1))
                        {
                        if(F1[i,j]==0)
                        {
                        Da=which(F1[,i]==1)
                        Db=which(F1[j,]==1)
                        D=expand.grid(Da,Db)
                        temp=0
                        if(nrow(D)>0)
                        {for (k in 1:nrow(D))
                        {
```

```
                        alpha=D[k,1]; beta=D[k,2]
                        temp=temp+P[j,beta]*P[alpha,i]*pi[alpha]
                        }}
                        ##print(temp)
                        change[i,j]=(pi[i]-temp)*P[i,j]
                                }
                        }
                }
                index=which(change==min(change),arr.ind=TRUE)
                F1[index[1],index[2]]=1
                A[[index[1]]]=setdiff(A[[index[1]]],c(index[2]))
                iter=iter+1
                tempsize=size(F1,P)
                print(tempsize)
                if(tempsize>=level)
                break
                F2=F1
        }
return(F2)
}
##root=compare(P)
##opt=optfilter(0.7,root,stage=3,P)
```

```
estimation=function(x,f)
{
#------------creating the observed chain-----
y=NULL
y[1]=x[1]

for (i in 1: (length(x)-2))

{
  if (f[x[i],x[i+1]]==1) ind1=1  else  ind1 =0

  if (f[x[i+1],x[i+2]]==1) ind2=1  else  ind2 =0

  if ( ind1==0 & ind2==0 )
y[i+1]= NA
else y[i+1]= x[i+1]
  i=i+1

}

if (f[x[i],x[i+1]]==1) ind1=1  else  ind1 =0

if ( ind1==0)  y[i+1]= NA  else y[i+1]= x[i+1]


#-checking if all the observed transitions occur in the chain---

index=which(f==1,arr.ind=T)

for ( k in nrow(index))
{
  i=index[k,1]
  j=index[k,2]
  ind=NULL
  for ( l in 1:(length(x)-1))
  {
    if (x[l]==i & x[l+1]==j)
    {
      ind[l]=1
      break
    }
  }
  if(all(ind==0)) stop("ERROR: All transitions are not there")
}

#------function to compute the matrix product-----
matprod=function(P,k)
{
  temp=diag(ncol(P))
  if(k>0)
  {
  for (i in 1:k)
    temp=temp%*%P
  }
  else if (k<0) stop("ERROR: k must be non-negative")
  else temp=temp     #------in case of P^0 the function returns I
  return(temp)
}

#-------function A,B to determine the states
#####and the number of  steps in a missing run-----
A=function(y,i) #-computes steps in runs of form a_ _ _ ..._ _
{
  temp=which(is.na(y)==FALSE)
  ind=min(temp[temp>i])
  if(ind==Inf)
  {
    step= length(y)-i
    state=0  #-----not actual state..used as a default indicator----
  }
  else
  {
    step=ind-i
    state=y[ind]
  }

  ans=list(state,step)
  names(ans)=c("state","steps")
  return(ans)
}

B=function(y,i)
{
  temp=which(is.na(y)==FALSE)
  ind=max(temp[temp<i])
  step=i-ind
  ans=list(y[ind],step)
  names(ans)=c("state","steps")
  return(ans)
}


#---------function determining the type of the transitions-----

type=function(y,i)
{
  if (is.na(y[i])!=TRUE & is.na(y[i+1])==TRUE) typ=1
  else if (is.na(y[i])==TRUE & is.na(y[i+1])!=TRUE) typ=2
  else if (is.na(y[i])==TRUE & is.na(y[i+1])==TRUE) typ=3
  else if (is.na(y[i])!=TRUE & is.na(y[i+1])!=TRUE) typ=4
  else typ=5

  return(typ)
}
```

```
#--------function which constructs P0 from a given P-----
nullmat=function(p)
{
  p0=p
  for (i  in 1:nrow(p))
  {
    for (j in 1:ncol(p))
    {
      if (f[i,j]==1) p0[i,j]=0
      else p0[i,j]=p[i,j]
    }
  }
  return(p0)
}

#--------function to determine p[alpha,beta]
######for each transition i-----

case1=function(p,i,y,alpha,beta)
{
  if (y[i]!=alpha)  estp=0
  else if (f[alpha,beta]==1) estp=0
  else
  {
    p0=nullmat(p)
    temp=A(y,i)
    b=temp$state
    k=temp$steps

    if(b!=0)
estp=p[alpha,beta]*(matprod(p0,(k-1))[beta,b])
/(matprod(p0,k)[alpha,b])
```

```
    else
estp= p[alpha,beta]*(sum(matprod(p0,(k-1))[beta,]))
/(sum(matprod(p0,k)[alpha,]))
  }

  return(estp)
}                  #------case 1 ends here

case2=function(p,i,y,alpha,beta)
{
  if(y[i+1]!=beta)  estp=0
  else if (f[alpha,beta]==1)  estp=0

  else
  {
    p0=nullmat(p)
    temp=B(y,(i+1))
    a=temp$state
    k=temp$steps
    estp=p[alpha,beta]*(matprod(p0,(k-1))[a,alpha])
/(matprod(p0,k)[a,beta])
  }

  return(estp)
}                  #--------case 2 ends here-----

case3=function(p,i,y,alpha,beta)
{
  if(f[alpha,beta]==1)   estp=0
 else
 {
  p0=nullmat(p)
  temp1=B(y,i)
  temp2=A(y,(i+1))
  a=temp1$state
  b=temp2$state
  m=temp1$steps
  n=temp2$steps
  k=m+n+1

  if(b!=0) estp=p[alpha,beta]*(matprod(p0,m)[a,alpha])
*(matprod(p0,n)[beta,b])/(matprod(p0,k)[a,b])

  else estp=p[alpha,beta]*(matprod(p0,m)[a,alpha])
*(sum(matprod(p0,n)[beta,]))/(sum(matprod(p0,k)[a,]))
 }
  return(estp)
}                # -----------case 3 ends here-----

case4=function(p,i,y,alpha,beta)
{
  if(y[i]==alpha & y[i+1]==beta)  estp=1

  else estp=0

  return(estp)
}               #-------case4 ends here-----

prob=function(p,i,y,alpha,beta)
{
  typ=type(y,i)
  if (typ==1)   estp=case1(p,i,y,alpha,beta)
  else if(typ==2)   estp=case2(p,i,y,alpha,beta)
  else  if(typ==3)  estp=case3(p,i,y,alpha,beta)
  else       estp=case4(p,i,y,alpha,beta)


  return (estp)
}

#--------------------------EM algorithm starts-----------------
dim_mat=nrow(f)
P1= matrix(rep(0.1,(dim_mat^2)),nrow=dim_mat)

P2=P1+0.5
temp1=P1
freq=P1
while(any(abs(P2-P1)>0.01))

{
  P1=temp1

  for(alpha in 1:nrow(P1))
  {
    for (beta in 1:ncol(P1))
    {
      su1=NULL
      obs=NULL
      for (k1 in 1: (length(y)-1))
      {
        su1[k1]=prob(P1,k1,y,alpha,beta)
      }
      freq[alpha,beta]=sum(su1)
    }
    print(sum(freq[alpha,]))
  }



  for(alpha in 1:nrow(P1))
  {
    for (beta in 1:ncol(P1))
    {
      P2[alpha,beta]=freq[alpha,beta]/sum(freq[alpha,])
    }
  }
  temp1=P2
print(P2)
}
return(P2)
}
```

```
completevar=function(Phat)
{
P2=Phat

#----function to find the expected
#------transitions given the data------

expectation=function(P)
{
  freq=P
  for(alpha in 1:nrow(P))
  {
    for (beta in 1:ncol(P))
    {
      su1=NULL
      for (k1 in 1: (length(y)-1))
      {
        su1[k1]=prob(P,k1,y,alpha,beta)

      }
      freq[alpha,beta]=sum(su1)
    }
  }
  return(freq)
}

#----------construction of B_i  matrices------

k=nrow(P)    #------number of states---
k2=k-1
B=NULL
freq=expectation(P2)

for ( i in 1:k)
{
  temp=matrix(rep(0,(k2*k2)),byrow=T,nrow=k2)

  for (j in 1:k2)
  {
    for(j1 in 1:k2)
    {
      if (j1!=j)
```

```
        temp[j,j1]= (freq[i,k])/(1*((1-sum(P2[i,1:k2]))^2))

      else
        temp[j,j1]=(1)*((freq[i,k]/((1-sum(P2[i,1:k2]))^2)
        +(freq[i,j]/(P2[i,j]^2))))
    }
  }

  B[[i]]=temp
}

require(Matrix)
bdiag(B)
ami=solve(bdiag(B))
return(ami)
}
```

```
sem=function()
{
source("D:\\Research2\\R codes\\Markov3.R")
##P2=estimation(x,f,k)
#-------take the estimated TPM (EM estimate)-------
Phat=P2
s=ncol(P2)
#------ EM estimate..we remove the last column because of restriction
theta_mat=P2[,-ncol(P2)]

theta=as.vector(t(theta_mat))
d=length(theta)

#-----function which computes E-step and M-step
#-------and returns next iterated estimate---
```

```
Em=function(Q)
{
  estp=Q
  for(alpha in 1:nrow(Q))
  {
    for (beta in 1:ncol(Q))
    {
      su1=NULL
      for (k1 in 1: (length(y)-1))
      {
        su1[k1]=prob(Q,k1,y,alpha,beta)

      }
      freq[alpha,beta]=sum(su1)
    }
  }

  for(alpha in 1:nrow(Q))
  {
    for (beta in 1:ncol(Q))
    {
      estp[alpha,beta]=freq[alpha,beta]/sum(freq[alpha,])
    }
  }
  return(estp)
}

#------------setting theta_t----
DM=matrix(rep(0,(d*d)),ncol=d)
for(i in 1:d)
{
P1=matrix(rep(0.1,9),nrow=3)
```

```
P2=P1+0.5
temp1=P1

R1=rep(0,d)
temp2=R1+2

while(any(abs(temp2-R1)>=0.001))
{
  index= which(abs(temp2-R1)<0.001)
  P1=temp1
  R1=temp2

    temp_P=P1[,-s]
    theta_t=theta
    theta_t[i]=as.vector(t(temp_P))[i]
    temp_P1=matrix(theta_t,byrow=T,ncol=(s-1))
    temp_P1=cbind(temp_P1,t(t(rep(1,s)-rowSums(temp_P1))))
    temp_P2=Em(temp_P1)
    temp_theta=as.vector(t(temp_P2[,-s]))

    for(j in 1:d)
    {
    temp2[j]=(temp_theta[j]-theta[j])/(theta_t[i]-theta[i])
    }


    for(j in index)   temp2[j]=R1[j]

  temp1=Em(P1)
  P2=temp1
  print(index)
  }
```

```
#print(temp2)
DM[i,]=temp2


}
return(DM)
}
```

R codes for Chapter 5

```
#------reading data file and basic exploratory analysis----
full_data=read.table("C:\\Users\\Public\\Documents
\\Sensor_data2.csv",header=T,sep=",")
data=full_data[,1]
#----convert continuous data into discrete chain


discretize=function(data)
{
        return(cut(data,br=quantile(data),labels=F))
}


full_chain=discretize(data)


##full_chain[13088]=1


#----- ML estimate of T.P.M based on complete data---


ML_ordinary=function(data)
{
        x=data
        dimension=length(unique(x))
        P <- matrix(nrow = dimension, ncol = dimension, 0)
        for (t in 1:(length(x) - 1))
        P[x[t], x[t + 1]] <- P[x[t], x[t + 1]] + 1
        for (i in 1:dimension) P[i, ] <- P[i, ] / sum(P[i, ])
        return(P)
}

complete_est=ML_ordinary(full_chain)

#------ partition full chain into two subsets:
#------ one subset for initial T.P.M and other subset to be filtered

train_chain= full_chain[1:5000]
test_chain=full_chain[-train_chain]
init_tpm= ML_ordinary(train_chain)

#------ choosing the filter matrix----
source("D:/research3.1R")
root=compare(init_tpm)
opt=optfilter(0.2,root,stage=3,init_tpm)

#------ filter the data and estimate the T.P.M
source("D:/Research2/R codes/estimationfunction.R")
fin_est=estimation(test_chain,opt)
```

Bibliography

1. *Kolkata case study*, https://trafficlogix.in (Retrieved online at 15/2/2022).
2. *Data never sleeps*, https://www.domo.com (Retrieved online at 15/2/2022), 2018.
3. *The growth in connected iot devices is expected to generate 79.4zb of data in 2025, according to a new idc forecast*, https://www.businesswire.com/ (Retrieved online at 15/2/2022), 2019.
4. T.W. Anderson and Leo A. Goodman, *Statistical inference about markov chains*, Annals of Mathematical Statistics **28** (1957), 89–109.
5. Peter J. Avery and Daniel A. Henderson, *Fitting markov chain models to discrete state series such as dna sequences*, Applied Statistics **48** (1999), 53–61.
6. Richard Baraniuk, *Compressive sensing*, IEEE Signal Processing Magazine **24** (2007), 118–121.
7. Richard Baraniuk, Mark Davenport, Ronald DeVore, and Michael Wakin, *A simple proof of the restricted isometry property for random matrices*, Constructive Approximation **28(3)** (2008), 253–263.
8. Richard G. Baraniuk, *More is less: Signal processing and the data deluge*, SCIENCE **331** (2011).
9. M.S. Bartlett, *The frequency goodness of fit test for probability chains*, Mathematical Proceedings of the Cambridge Philosophical Society **47** (1951), 86–95.
10. Patrick Billingsley, *Statistical methods in markov chains*, The Annals of Mathematical Statistics **32(1)** (1961), 12–40.
11. Emamnuel J. Candes, *Compressive sampling*, Proceedings of the International Congress of Mathematics (2006), 1–20.

12. Emamnuel J Candes, J K Romberg, and T Tao, *Stable signal recovery from incomplete and inaccurate measurements*, Communications on Pure and Applied Mathematics **59** (2006), no. 8, 1207–1223.

13. Emamnuel J. Candes and Michael B. Wakin, *An introduction to compressive sampling*, IEEE Signal Processing Magazine **25(2)** (2008), 21–30.

14. Emmanuel J. CandÃšs, Justin Romberg, and Terence Tao, *Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information*, IEEE TRANSACTIONS ON INFORMATION THEORY **52** (2006), no. 2, 489–509.

15. G. Casella and E.L. Lehman, *Theory of point estimation*, Springer, 2003.

16. W. G. Cochran, *Sampling techniques*, Wiley, 1963.

17. Bruce A. Craig and Peter P. Sendi, *Estimation of the transition matrix of a discrete-time markov chain*, Health Economics **11** (2002), 33–42.

18. AP Dempster, NM Laird, and DB Rubin, *Maximum likelihood from incomplete data via the EM algorithm*, Journal of the Royal Statistical Society, Ser. B **39** (1977), 1–38.

19. David.L. Donoho, *Compressed sensing*, IEEE Transactions On Information Theory **52** (2006), no. 4, 1289–1306.

20. J. L. Doob, *Stochastic processes*, John Wiley and Sons, New York, 1953.

21. Marco F. Duarte, Mark A. Davenport, Dharmpal Takhar, Jason N. Laska, Ting Sun, Kevin F. Kelly, and Richard G. Baraniuk, *Single-pixel imaging via compressive sampling*, IEEE Signal Processing Magazine **25(2)** (2008), 83–91.

22. R.M. Fano, *The transmission of information*, Technical Report, Research Laboratory of Electronics at MIT (1949), no. 65.

23. Atanu Kumar Ghosh and Arnab Chakraborty, *Use of em algorithm for data reduction under sparsity assumption*, Computational Statistics **32** (2017), no. 2, 387–407.

24. James Glanz, *Power, pollution and the internet*, The New York Times (Retrieved online at 15/2/2022), 2012.

25. Alfred Haar, *Zur theorie der orthogonalen funktionensysteme*, Mathematische Annalen **69** (1910), 331–371.

26. W. B. Hocking, R. R. & Smith, *Optimum incomplete multi-normal samples*, Technometrics (1972).

27. IRE, *A method for the construction of minimum redundancy codes*, vol. 40, 1952.

28. Uthayakumar Jayasankar, Vengattaraman Thirumal, and Dhavachelvan Ponnurangam, *A survey on data compression techniques: From the perspective of data quality, coding schemes, data type and applications*, Journal of King Saud University - Computer and Information Sciences **33** (2021), no. 2, 119–140.

29. Oscar Kempthorne, *The deign and analysis of experiment*, Wiley, 1952.

30. D K Kim and J M G Taylor, *The restricted EM algorithm for maximum likelihood estimation under linear restrictions on the parameters*, Journal of the American Statistical Association **90** (1995), no. 430, 708–716.

31. E.L. Lehman, *Testing statistical hypothesis*, Wiley, 1959.

32. R. J. A. Little and D. B. Rubin, *Statistical analysis with missing data*, John Wiley, 1987.

33. Gene H. Golub & Charles F. Van Loan, *Matrix computations*, The John Hopkins University Press, 2013.

34. T. A. Louis, *Finding the observed information matrix when using the EM algorithm*, Journal of the Royal Statistical Society, Ser. B **44** (1982), 226–233.

35. S. Mallat, *A wavelet tour of signal processing*, Academic Press, 1999.

36. Patrick E. McKnight, Katherine M. McKnight, Souraya Sidani, and Aurelio Jose Figueredo, *Missing data: A gentle introduction*, THE GUILFORD PRESS, 2007.

37. G. J. McLachlan and T. Krishnan, *The em algorithm and extensions*, John Wiley, 2008.

38. Fabrizia Mealli and Donald B. Rubin, *Clarifying missing at random and related definitions, and implications when coupled with exchangeability*, Biometrika **102** (2015), no. 4, 995–1000.

39. X. Meng and D. B. Rubin, *Using EM to obtain asymptotic variance-covariance matrices: The SEM algorithm*, Journal of the American Statistical Association **86** (1991), no. 416, 899–909.

40. Justin Romberg, *Imaging via compressive sampling*, IEEE Signal Processing Magazine **25(2)** (2008), 14–20.

41. Donald B. Rubin, *Inference and missing data*, Biometrika **63** (1976), no. 3, 581–592.

42. Claude E. Shannon, *A mathematical theory of communication*, Bell Telephone Systems Publication, Monograph B-1598 **27** (1948), 379–423, 623–656.

43. C.E. Shanon, *Communications in the presence of noise*, Proc. IRE **37** (1949), 447–457.

44. N. Z. Shi, S. R. Zheng, and J. Guo, *The restricted EM algorithm under inequality restrictions on the parameters*, Journal of Multivariate Analysis **92** (2005), 53–76.

45. M. Tan, G.L. Tian, and H.B. Fang, *Estimating restricted normal means using the em-type algorithms and ibf sampling*, World Scientific, New Jersey, 2003.

46. G. L. Tian, K. W. Ng, and M. Tan, *EM-type algorithms for computing restricted MLEs in multivariate normal distributions and multivariate t-distributions*, Computational Statistics and Data Analysis **52** (2008), 4768–4778.

47. R. E. Trawinski, I. M. & Bargmann, *Maximum likelihood estimation with incomplete multivariate data*, Annals of Mathematical Statistics (1964).

48. Terry Welch, *A technique for high-performance data compression*, IEEE Computer **17** (1984), 8–19.

49. Rebecca M. Willett, Roummel F. Marcia, and Jonathan M. Nichols, *Compressed sensing for practical optical imaging systems: a tutorial*, Optical Engineering **50(7)** (2011), 072601–1 – 072601–12.

50. J. Ziv and A. Lempel, *A universal algorithm for sequential data compression*, IEEE Transactions on Information Theory **IT-23** (1977), no. 3.