Analysis of Two Dimensional Degraded Signals

By

Azka Maqsood

A thesis submitted to the faculty of Electrical Engineering Department, Military College of Signals, National University of Science and Technology, Rawalpindi in fulfillment of the requirements for the degree of MS in Electrical Engineering.

October 2017

SUPERVISOR CERTIFICATE

It is to certify that the final copy of the thesis has been evaluated by me, found as per the specified format and error free.

Date: _____ _____

Col. Dr. Imran Tauqir

ABSTRACT

Wavelet based image processing techniques do not strictly follow the conventional probabilistic models that are unrealistic for real world images. However, the key features of joint probability distributions of wavelet coefficients are well captured by Hidden Markov Tree model.

This thesis presents Hidden Markov Tree model based technique consisting of Wavelet based Multiresolution analysis to enhance the results in image processing applications such as compression, classification and denoising. The proposed technique is applied to colored video sequences by implementing the algorithm on each video frame independently. A 2D-Discrete Wavelet Transform is used which is implemented on popular Hidden Markov Tree Model used in the framework of Expectation Maximization algorithm. The proposed technique can properly exploit the temporal dependencies of wavelet coefficients and their non-Gaussian performance as opposed to existing wavelet based denoising techniques which consider the wavelet coefficients to be jointly Gaussian or independent.

Denoised frames are obtained by processing the wavelet coefficients inversely. Detailed comparison has been made with the existing state of the art techniques.The proposed denoising method reveals improved results in terms of quantitative and qualitative analysis for both additive and multiplicative noise and retains nearly all the structural contents of a video frames.

DEDICATION

I dedicate this report to my beloved parents, and my supervisor, Col. Dr. Imran Touqir for their prayers and encouragement.

ACKNOWLEDGMENT

I thank Allah who provided me with strength and caliber to bring this thesis work to its successful completion.

I am deeply obliged to my supervisor, Col. Dr. Imran Tauqir, for his guidance, unwavering support and confidence in me throughout the course of this thesis work. His time and efforts were very valuable. He contributed significantly in the thesis work and also imparted a lot of knowledge to me. I am also grateful to my Guidance and Evaluation Committee, Lt. Col. Dr. Adil Masood, and Lt. Col. Dr. Adnan Rashdi for dedicating their time and making contributions to this thesis work. I offer my regards and blessings to all of those who supported me in any respect during the completion of this work. Alongside that, I thank the university administration, for facilitating the progress of work at different phases, faculty members of MCS for polishing my knowledgebase, my university mates and friends.

Last but not the least; I am grateful to those people without whom I could never have accomplished my aim my family, especially my parents. I am grateful to them for their constant prayers and unfaltering support throughout this journey. May Allah bless them all with eternal happiness.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

1.	Mean Square Error	**MSE**
2.	White Gaussian Noise	**WGN**
3.	Additive White Gaussian Noise	**AWGN**
4.	Continuous Wavelet Transform	**CWT**
5.	Discrete Wavelet Transform	**DWT**
6.	Expectation Maximization	**EM**
7.	Gaussian Mixture Model	**GMM**
8.	Hidden Markov Model	**HMM**
9.	Probability Density Function	**Pdf**
10.	Hidden Markov Tree	**HMT**
11.	Probability Mass Function	**pmf**
12.	Peak Signal to Noise Ratio	**PSNR**
13.	Maximum a posterior	**MAP**
14.	Inverse Discrete Wavelet Transform	**IDWT**

CHAPTER 1

INTRODUCTION

Denoising can effectively enhance visual quality and considerably simplify the subsequent processing tasks like video compression and pattern recognition. Video denoising considers time-frequency data of a video signal and it is different from image denoising. It can be achieved by different approaches: Time-domain, Frequency-domain, and Time-Frequency combination.

Frequency domain methods of denoising are limited in their scope as these methods do not take temporal correlation between frames into account [1][2][3]. Wiener filter is an example of spatial filter that removes spatial noise from images and succeeded in achieving high gain. However this filter cannot restore edges especially in less noisy areas [4].

Time domain methods consider the inter-frame correlation between frames and perform well for still videos without motion [5]. In case of videos having motion, these methods do not provide significant results. Rakhshanfar in [6] proposed a temporal filter for denoising of frames that provide considerably good results in noise removal process and produce less blocking artifacts but it causes blurring effect.

Yun Liu and Bing Luo in [7] introduced a method based on total variation (TV) and temporal filtering for image denoising. The temporal filter maintains structure and edges well but it cannot reduce noise. The TV algorithm is applied on a noisy frame to reduce noise but it could not restore structure information.

Time-frequency methods consider both spatial and temporal correlations between different frames in a video sequence and provide efficient results. Transform domain methods are example of Time-frequency methods and these methods exploit the sparsity of data and has good localization properties and multiresolution characteristics in either time domain or frequency domain. These properties makes it more useful to separate a useful signal from noise. Hence, wavelet has gained popularity for image denoising [10][11]. Wavelet transform can be 2D or 3D. 3D transform domain methods do not perform well for denoising purpose because of long delay and inability to adapt to fast motions in a video sequence [12].

Di Zhigang et al. [13] proposed wavelet based threshold function to overcome the discontinuity of hard threshold method and soft threshold method at threshold value.

Probabilistic Graphical Models designed in time-frequency domain are used for denoising [14]. Hidden Markov Model (HMM) with 2D Discrete Wavelet Transform (DWT) is utilized to get efficient results.

Jinn and Hwang proposed image noise reduction method through the wavelet domain Bayesian threshold criterion coefficient of shrinkage method [15].Techniques like VBM3D [16] and E-RF3 [17] have been the most efficient ones in denoising as they exploit DCT in their framework.

1.1 Research already carried out

Statistical signal processing has a wide range of applications and is taught at graduate level in Electrical Engineering, Applied Mathematics and Statistics. Time Frequency analysis of signal processing treats signals as stochastic processes, dealing with their statistical properties like mean, variance etc.

Signal processing deals with the analysis of signals that are random in nature and processing them using some statistical techniques. Many real time signals have a structure or a component that is stochastic in nature. These signals are subjected to unwanted noise quite often and there is a need to model them in stochastic domain. For every signal that is compressed, the theory of compression deals it in the form of probabilistic model. The reason is that, the signal is random in nature and these techniques play a vital role in evaluating such signal.

1.2 Thesis Statement

Most of the existing spatio-temporal methods can effectively remove uniform noise from color images and video sequences but do not perform well for speckle noise.

In this work, a combined spatial and temporal filtering based noise removal algorithm is proposed that can remove Gaussian as well as speckle noise from color image and video sequences considerably. The proposed technique gives efficient results for de-speckling of

images as well. The results shows that the proposed method do not remove noise only but also retains almost all the structural information of a video frame.

1.3 Objective

The primary objective is to develop a model based on time-frequency analysis of a two dimensional signals. This framework allows to model non-Gaussian statistics of individual wavelets coefficients and exploits the dependences between these coefficients. In this model, efficient expectation maximization algorithm is applied to probabilistic graphical models that enable to achieve compression of signals. Time-frequency analysis combined in probabilistic graphical models; provide modeling of data for multiresolution.

1.4 Methodology used

The methodology used in this thesis is divided into two steps.

- First step is to develop a model in wavelet domain. Initially, we will focus on achieving the primary properties of wavelet transform. And then this framework will be extended with the use of probabilistic graphical models to obtain the secondary properties of wavelet transform.
- Second step will be the development of algorithm that is to achieve de-noising of two dimensional color video signals.

1.5 Advantages

i) Efficient in terms of computational complexity.

ii) Due to reduction in computational complexity proposed framework will be efficient for:

- Signal De-noising
- Signal Classification
- Signal Detection
- Signal Estimation
- Signal Compression

1.6 Areas of Application

This methodology has its wide range of applications in;

- Image Processing in computers, digital cameras and imaging systems
- Video Processing for interpreting moving pictures
- Speech Signal Processing for processing and interpreting spoken words
- Audio Signal Processing for representing electrical signals as sound, such as speech or music
- Array Processing for processing signals from array of sensors
- Wireless Communications such as; waveform generation, demodulation, filtering, equalization Control Systems
- Feature Extraction like speech recognition and image understanding
- Compression techniques such as Video Compression, Image Compression and Audio compression

1.7 Outline of Thesis

This thesis is based on following chapters

Chapter 01:

This chapter is the basic introduction of the topic, problem statement, scope and objective.

Chapter 02:

Chapter 02 will be introduction of wavelet basis and transform. This chapter will give the basic understanding of techniques used in statistical signal processing and their importance.

Chapter 03:

It deals with brief explanation of probabilistic models used for image modeling and processing. This chapter explains the structure of models and their calculations in detail.

Chapter 04:

Chapter 04 discusses the Gaussian Models and their different types.

Chapter 05:

This chapter is about the Hidden Markov model and its framework in terms of Gaussian mixture models.

Chapter 06:

Chapter 06 discusses the Statistical Image Models in Time-Frequency Domain that helps in achieving the de-noising of signals.

Chapter 07:

This chapters presents different results and simulation.

CHAPTER 2

Analysis of Signals in Time-Frequency Domain

Time frequency analysis basically deals with the analysis of the objects at different scales of resolution that are small or large in size with low or high contrast. This is commonly stated as Multiresolution processing/analysis (MRA). Through MRA, a signal can be analyzed at different frequencies with different resolutions. Images are 2D arrays containing edges and smooth regions with different varying statistics.

Since 1950's the Fourier transform was the backbone of transform-based image processing but it is much easier to transmit, compress and analyze the two dimensional signal by using wavelet transform. Wavelet transform is based on wavelets (small waves) as opposed to Fourier transform whose basis functions are sinusoids. Unlike Fourier transform that only provide the frequency information of a signal, both time and frequency information of a signal is obtained by wavelet transform, which helps in better analysis of a signal.

The transform is computed for different scales of wavelet and at various locations of signal, thus the plane of transform is filled. On carrying out the process in a continuous and smooth way (i.e., the position and scale is smoothly varied) such transform is called continuous wavelet transform (CWT). If the position and scale change discretely, the transform is called discrete wavelet transform (DWT).

2.1 Continuous Wavelet Transform

To overcome the resolution problem, an alternate approach known as continuous wavelet transform (CWT) was developed. CWT is similar to Short Term Fourier Transform (STFT) as in both the signal is multiplied by a function but unlike STFT the Fourier transform of windowed signal is not taken and variable size window is used for each spectral component. One illustration is that the negative frequencies are not computed as in the case of Fourier Transform. The continuous wavelet transform offers better time and frequency localization by constructing the time-frequency representation of a signal.

The continuous wavelet transform of any signal $z(t)$ is given as:

$$Z_w(\tau, s) = \frac{1}{\sqrt{s}} \int_{-\infty}^{\infty} z(t) \psi^*{}_{\tau,s} \left(\frac{t-\tau}{s} \right) dt \tag{2.1}$$

and

$$\psi_{\tau,s} = \frac{1}{\sqrt{s}} \psi \left(\frac{t-\tau}{s} \right) \tag{2.2}$$

where tau and s are the translation and scaling parameters respectively. psi(t) is the transforming function and known as mother wavelet.

To reconstruct the original signal back, inverse CWT can be applied as:

$$z(t) = \frac{1}{M_\psi^2} \int_s \int_\tau Z_w(\tau, s) \frac{1}{s^2} \psi \left(\frac{t-\tau}{s} \right) d\tau \, ds \tag{2.3}$$

where M_psi is constant known as admissibility constant that depends on wavelet used. The following condition should meet for successful reconstruction:

$$M_\psi = \left\{ 2\pi \int_{-\infty}^{\infty} \frac{|\widehat{\psi}(\xi)|^2}{|\xi|} d\xi \right\}^{\frac{1}{2}} < \infty \tag{2.4}$$

Morlet wavelet and the Mexican hat wavelet are the two examples of CWT. Discretized version of CWT gives us wavelet series and it takes much time for the highly redundant information to process during reconstruction of a signal. On the other hand, Discrete wavelet transform (DWT) has less computational complexity and it provides appropriate information for synthesis and analysis of actual signal.

2.2 Discrete Wavelet Transform

The basic idea of CWT and DWT is same which focus on time-scale representation of a digital signal. The CWT is computed if we change the scale of the window, shift it in time, multiply it by the signal, and integrate over all times. While in DWT, different filters with different cutoff frequencies are used for the signal analysis at different scales.

The signal to be analyzed is passed through a sequence of filters consisting of high pass and low pass filters to analyze the high pass and low pass frequencies respectively. The filtering operation will change the signal resolution and the scale will be changed by up sampling (down sampling) operations. Down sampling refers to omit the every other sample of a signal and in up sampling, sampling rate is enlarged by adding new samples to the signals.

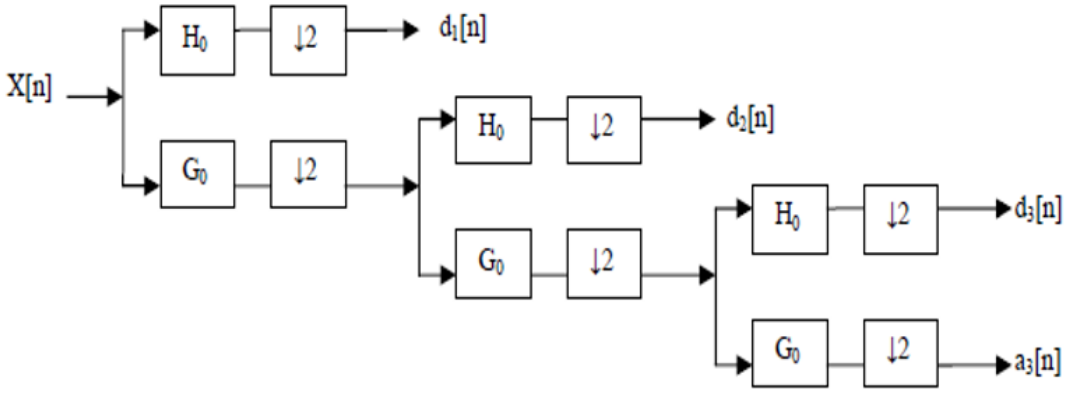


Fig 2.1 Three level Wavelet Decomposition

In the above figure, $X[n]$ is the input signal and n is an integer. G_o and H_o are the low pass and high pass filters. At each level, the high pass filter provide detail information of the input signal; denoted by d[n], whereas the low pass filter is responsible for scaling function that gives approximations of the signal, denoted by a[n].

The filtering process accompanied by the down sampling (decimation) continues until the desired level is reached. The length of the signal determines the number of levels.

2.2.1 Perfect Reconstruction

Reconstruction is opposite to the process of decomposition shown in diagram 2.2 below. The approximation and detail coefficients are up sampled, passed through high and low pass filters and added together to get the original or reconstructed signal. G_1 and H_1 as synthesis filters.

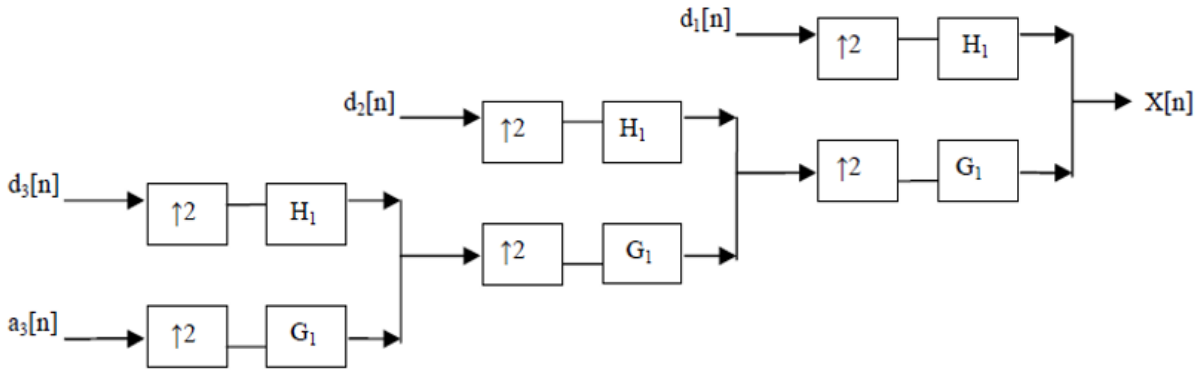


Fig. 2.2 Three Level Wavelet Reconstruction

2.2.2 Conditions for Perfect Reconstruction

There is a need for analysis and synthesis filters to satisfy some conditions in order to achieve perfect reconstruction. Let $G_o(z)$ denote the low pass analysis filter and $G_1(z)$ as low pass synthesis filters whereas, $H_o(z)$ and $H_1(z)$ are the high pass analysis and synthesis filters respectively. These filters have to satisfy the following conditions given in order to get perfect reconstruction.

$$G_o(-z)G_1(z) + H_o(-z)H_1(z) = 0 \tag{2.5}$$

$$G_o(z)G_1(z) + H_o(z)H_1(z) = 2z^{-d} \tag{2.6}$$

The accuracy of perfect reconstruction can be checked through different parameters like Peak Signal to Noise Ratio (PSNR) and Structural similarity index (SSIM). Some applications do not require reconstruction like pattern recognition and for such applications; the above conditions need not be applied.

2.3 Wavelet Families

The efficiency in using wavelet transform comes from the fact that right type of mother wavelet is chosen to encounter the details about a certain application. We can find plenty of wavelet basis functions used as a parent wavelet for wavelet transformation.

A parent wavelet is responsible for determining the characteristics of a certain wavelet transform as the parent wavelet results in the production of wavelet functions by dilations and translations. A significant amount of contribution has been experienced by the area of wavelets from Daubechies. Hence, in this work Daucechies 8 wavelet has been used to perform DWT.

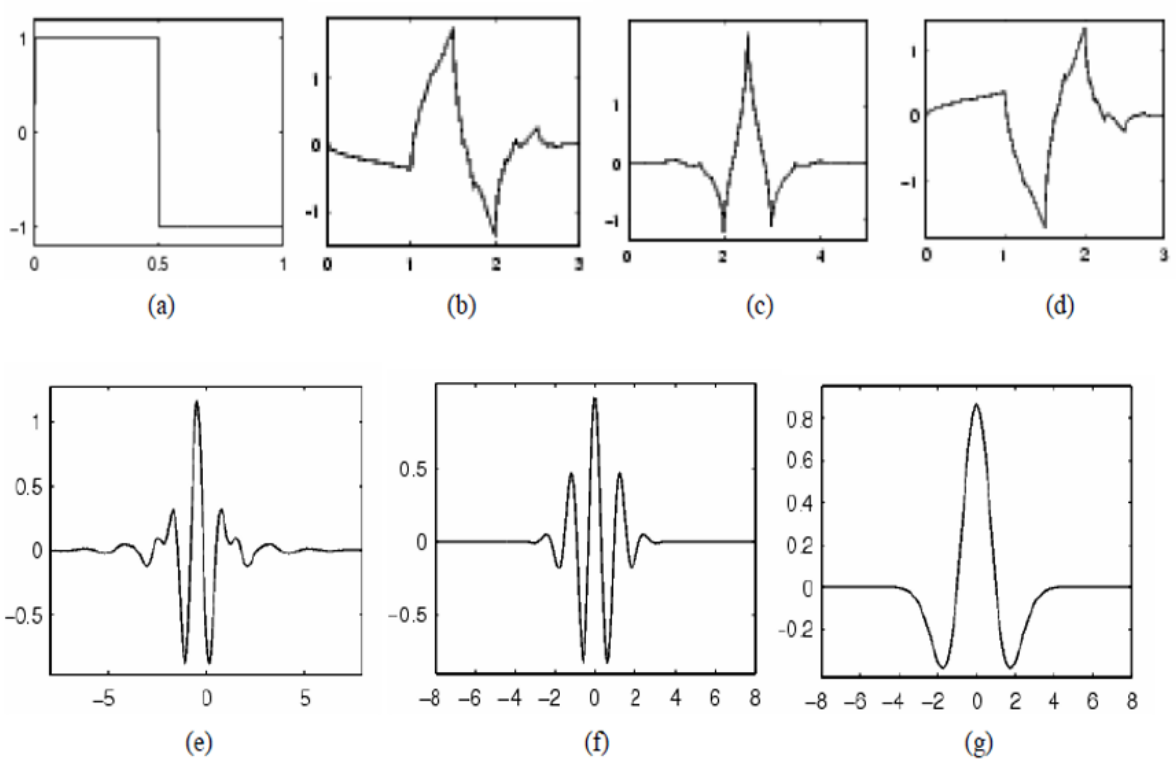


Figure 2.3 Wavelet Families (a) Haar (b) Daubechies (c) Coiflet1 (d) Symlet2 (e) Meyer (f) Morlet (g) Mexican Hat

CHAPTER 3

Modeling Processes

3.1 What is a model?

Before we entered into the details of how a model works, it is important to discuss what a model is. There are two different views about the meaning of model. One view is *mechanistic* one in which model interpret the mechanism by which something happens. These models are difficult to create often requiring year of experimental work but they are very powerful. A different view of models considers them as *black boxes*. In this approach, a model is evaluated on the basis of its accuracy in prediction, not by the mechanism used. To make numerically accurate and completely mechanistic models is rarely possible in real world task.

When there is the need of analyzing and modelling a database of sequences, the predictions that can be obtained by a black-box model are somewhat limited. The models we will focus on, namely *Hidden Markov Models*, fall somewhere between the extremes of mechanistic models and pure black-box models. These models do not provide mechanistic explanations but they have an internal structure that can provide an insight into the characteristic dynamics of the modeled process. We can easily modify this kind of structure according to our domain knowledge in order to improve the model performances.

From a general point of view, a model can be used for three main purposes: describing the details of a process, predicting its outcomes, or for classification purposes. All these three tasks cannot be performed by all the models. When there is the need of modelling processes that are characterized by great complexity and are affected by randomness, usually the main approach is to focus only on the aspects required for solving the task; in this way we can control the computational complexity of the models, in such a way to obtain practically useful ones.

3.2 Generative and Discriminative Models

Generative models completely describe the data, while discriminative models describe only the differences between classes without considering the classes themselves. Given a vector of features y and a finite set K of classes to which this vector may belong, a discriminative model describes the conditional probability $p(k|y)$ where $k \in K$, while a generative model represents the joint probability $p(k, y)$.

To understand the difference between those two categories it could be helpful considering a scenario in which we are collecting different set of sequences of system calls generated by k different processes. Each sequence is represented by means of a features vector y and it is labelled according to its class. If our objective is, for a new sequence y_i, to determine which process it belongs to (which process most probably generated it under certain assumptions), the natural choice is to use a discriminative model. Discriminative approach introduced a parametric model for the posterior probabilities, and a set of labelled training data is used to infer the values of parameters. From basic decision theory that says that the complete characterization of the solution can be explained by the set of posterior probabilities $p(k|y_i)$. Once we find these probabilities, it is easy to allocate the new sequence to a particular process. In a generative approach we model the joint distribution $p(k, y_i)$ of sequences and labels. This can be achieved by learning prior probabilities p(k) of class and the conditional densities for each class $p(y_i|k)$. Afterward, applying the Bayes theorem in order to compute the posterior probabilities:

$$p(k|y_i) = \frac{p(y_i|k)p(k)}{\sum_{j \in K} p(y_i|j)p(j)} \tag{3.1}$$

$p(k|y_i)$ is always possible to compute when $p(y_i|k)$ and $p(k)$ are given by using Bayes Theorem. But the discriminative approach is typically better if our objective is to only discriminate among classes. It is in fact, difficult to model the full joint distribution of a class when the data is highly structured; in this case we need a lot of examples in order to characterize them. Apart from this drawback, discriminative models are usually very fast at assigning new data to a class, while generative models generally need iterative solutions.

3.3 Probabilistic Graphical Model

Probabilistic graphical model is a tool that allows the problems of uncertainty and complexity to be dealt with in a natural way. These models can be seen as a merge between probability theory and graph theory. They are playing an increasing role in Machine Learning, because they are based on well-studied classical multivariate probabilistic systems, and at the same time, the graph theoretic side of graphical models provides both an edge by which humans are capable of modelling highly interacting set of variables, as well as a data structure that is well suited to design general-purpose algorithms. *Hidden Markov Model* can be seen as a special kind of graphical models.

From an abstract point of view, a graphical model is a statistical model, where the joint distribution P_θ is expressed by means of an underlying graph. The nodes of graph represent random variables and edges represent probabilistic relationships between variables. The idea is to represent a complex distribution involving a (possibly) large number of random variables as a product of local functions, where each variable depends only on a small number of related variables, according to the specific independence assumptions that have been done.

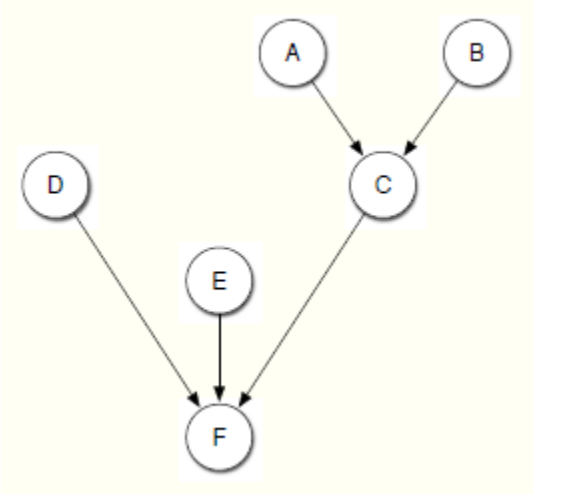


Fig 3.1 Representation of graphical model with 6 states

3.3.1. Types of Probabilistic Graphical Model

There are basically two branches of graphical representations namely Bayesian networks and Markov networks. Generally, probabilistic graphical models use a representation that is graph-based for encoding the complete distribution over a multidimensional space. This graph is referred as a compact representation of the group of independencies that are incorporated within the underlying specific distribution. Both families of graphical models cover the properties related to factorization and independences. The difference lies in the way they encode the independences and persuade factorization within the distribution.

Whenever there is a need for modeling such variables which have conditional dependence on one another, Directed graphs are a suitable choice. This phenomenon is commonly known as progression of events that have temporal relationship. On the contrary, undirected models are appropriate in modeling such data in which such relationship does not exist.

3.3.2. Directed and Undirected Models

Directed graphs are known as Bayesian Networks, and undirected graphs are Markov Random Fields. They have different properties with different advantages, but the crucial difference is in the definition of conditional independence. Undirected graphs are really flexible because they allow potential functions that are not probability distributions, but they are also difficult to be applied to big-sized task. The reason is due to the high computational cost in computing the normalizing constant. Actually the only algorithms that could be used efficiently to perform this task are approximate ones.

In general, directed and undirected graphs make different declarations of conditional independence; then we have families of probability distributions captured by a directed graph which are not captured by undirected graphs and vice versa. Two examples are given below.

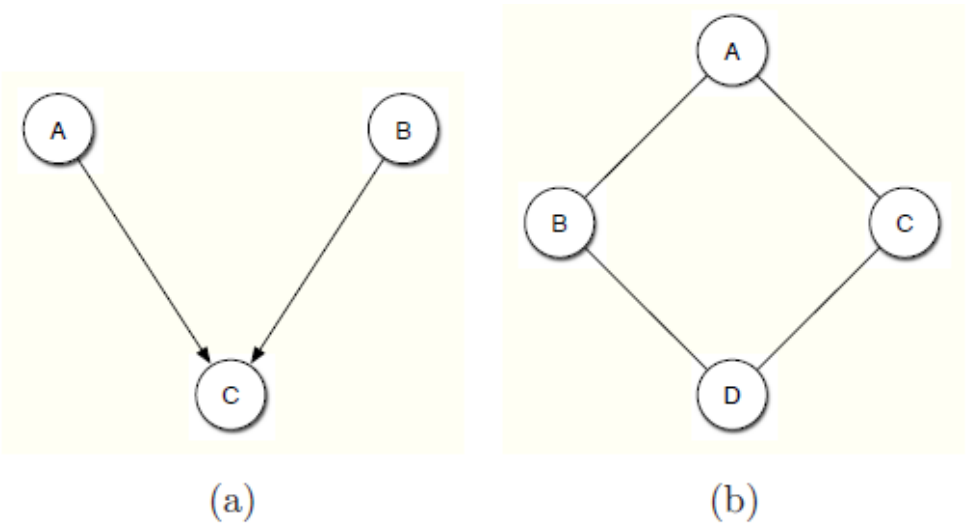


Fig 3.2 An example of directed (a) graph that cannot be re-expressed as undirected graph and (b) vice versa.

3.3.3 Representing Multivariate Distribution

Probabilistic Graphical Models are mainly used to provide more intuitive tools for dealing with multivariate probabilistic models. Such models are represented by joint probability of its variables $P\ (Y_1, Y_2, \dots, Y_n)$.

Probabilistic Graphical Model is to represent such joint probabilities in terms of conditional probabilities. It can be rewritten as:

$$P(Y_1, Y_2, \dots, Y_n) = P(Y_1)P(Y_2|Y_1)P(Y_3|Y_1, Y_2), \dots \dots, P(Y_n|Y_1, Y_{n-1}) \qquad (3.2)$$

The above equation assumes no prior independency information on data. For the case of complete independency of models random variables, the joint probability is defined as

$$P(Y_1, Y_2, \dots, Y_n) = P(Y_1)P(Y_2)P(Y_3) \dots P(Y_n) \qquad (3.3)$$

3.3.4 Markov Networks

Markov Networks belongs to a family of Undirected Graphs. Markov networks are simplest model that becomes a suitable choice when there is a need to model the data termed as discrete process in which the future values of the system depends only on the actual state of the system and does not have any dependence on the past states of the system.

Lets assume that there are two events that causes the grass to be wet: one can be sprinkler and second is raining. Also, suppose that whenever it is, the sprinkler is usually not turned on.

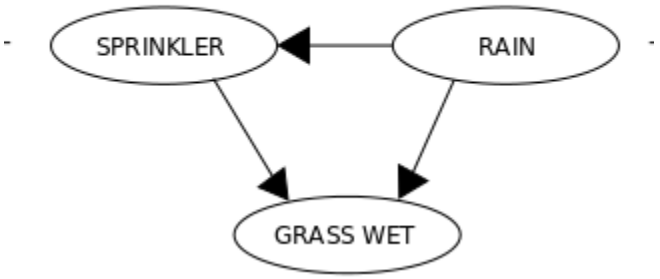


Fig 3.3 Example of Markov process

We can see that this graph is fully connected from one node to another.

3.3.5 Conditional Independence for Markov network

A node y_i is conditionally independent of all other nodes in the network given its Markov Blanket that is the set of all the neighbors of yi.

3.3.6 Bayesian Networks

A Bayesian Network (BN) is a probabilistic graphical model that represents a probability distribution through a directed acyclic graph (DAG) that encodes conditional dependency and independency relationships among variables in the model. In this specific graph structure, nodes

represent random variables, and each directed edge represents a dependency relationship between the two variables to which it connects.

Now if we suppose that the sprinkler's rain detection system is broken, and it can no longer tell when it's raining or not. The above scenario will be changed as:

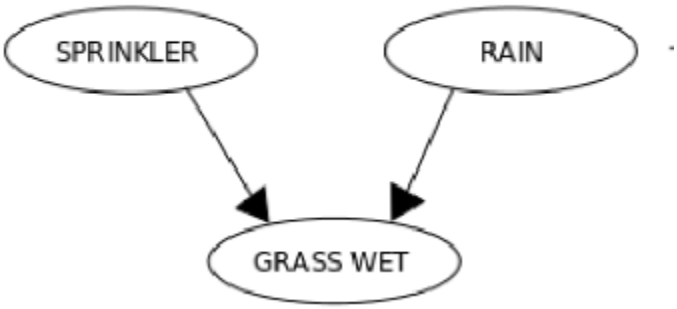


Fig. 3.4 Example of Bayesian Network

Now the sprinkler's activation does not depend on the rain. The biggest benefit here is independencies encoded in Bayes network.

CHAPTER 4

Gaussian distribution

Gaussian distribution is also referred to as Normal distribution, which is symmetric, continuous and bell-shaped distribution of a variable. This distribution is useful because of central limit theorem.

Central Limit Theorem

Central limit theorem says that for any sequence having number of trials, the standardized sample mean approaches the standard normal variable as number of trial increases.

$$P(a \leq Z_n \leq b) \approx P(a \leq \phi \leq b) \tag{4.1}$$

Where Z_n denotes standardized sample mean and ϕ is standard normal distribution.

In other words, as the number of independent random variables becomes very large say n→∞, they start following Normal distribution.

4.1. Univariate case

Gaussian distribution is bell shaped distribution and in case of single variable its pdf is written as,

$$p(x) = N(\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \qquad (-\infty < x < \infty) \tag{4.2}$$

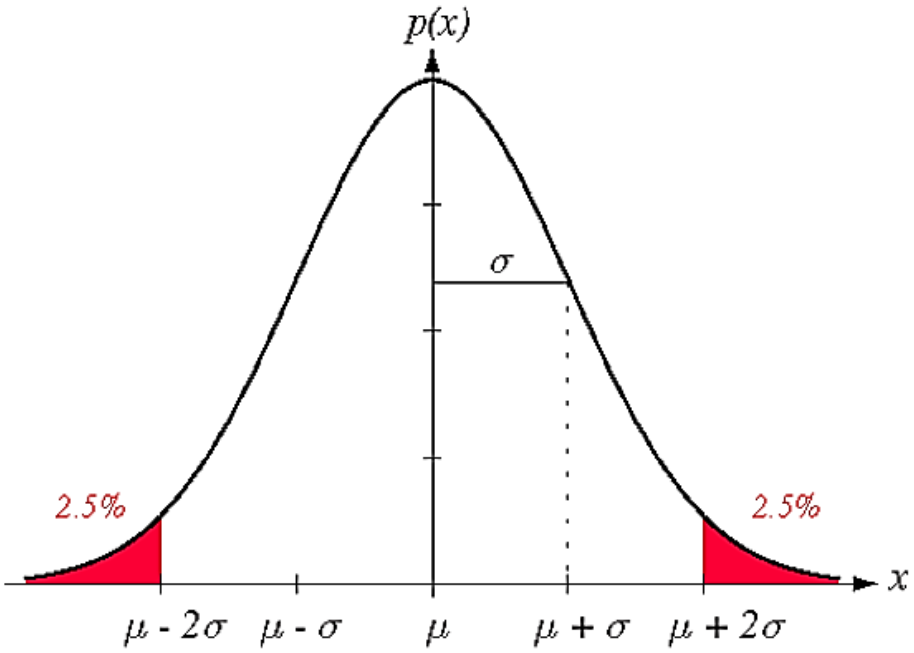


Fig 4.1: Illustration of Univariate Gaussian distribution in one-dimensional space.

A univariate Gaussian distribution has roughly 95% of its area in the range $|x - \mu| \leq 2\sigma$.Gaussian distribution is characterized by μ which is mean and σ^2 is variance which shows deviation from the mean value.

Where,

$$\mu = E(x) = \int_{-\infty}^{\infty} x\, p(x)\, dx \tag{4.3}$$

$$\sigma^2 = E(x - \mu)^2 = \int_{-\infty}^{\infty} (x - \mu)^2 p(x)\, d(x) \tag{4.4}$$

4.2 Bivariate case

It is easy to extend the case of one dimension to the higher dimensions with a K-dimensional vector variable X with mean vector μ and covariance matrix Σ.

For the bivariate case K=2 which means we have two random variables X_1 and X_2 then $X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$, $\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$ and $\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$. For the two dimensional vector X the bivariate takes the form:

$$p(x_{1,}\, x_2) = \frac{1}{(2\pi)|\Sigma|^{\frac{1}{2}}} e^{\left(-\frac{1}{2}\right)\left[\left(\frac{x_1-\mu_1}{\sigma_1}\right)^2 + \left(\frac{x_2-\mu_2}{\sigma_2}\right)^2\right]} \tag{4.5}$$

4.2.1 Case 1

When two variables are independent of each other $\sigma_{12} = \sigma_{21}$ and $\sigma_{12} = 0$ we have

$$f(x_1, x_2) = f(x_1)f(x_2) \;= \frac{1}{2\pi\, {\sigma_1}^2 {\sigma_2}^2} e^{-\left(\frac{1}{2}\right)\left[\left(\frac{x_1-\mu_1}{\sigma_1}\right)^2 + \left(\frac{x_2-\mu_2}{\sigma_2}\right)^2\right]} \tag{4.6}$$

$$|\Sigma| \;=\; \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{21} & \sigma_2^2 \end{bmatrix} = \sigma_1^2\, \sigma_2^2 - \sigma_{12}\sigma_{21} \tag{4.7}$$

$\because\ \sigma_{12=}\sigma_{21}$

As we assumed $\sigma_{12} = 0$,

$$|\Sigma| = \; \sigma_1^2\, \sigma_2^2 \tag{4.8}$$

If we take square root of the above equation,

$$|\Sigma|^{\frac{1}{2}} = (\sigma_1^2 \sigma_2^2)^{\frac{1}{2}} \tag{4.9}$$

Hence we can write constant part of bivariate equation as

$$f(x_1, x_2) = \frac{1}{2\pi^{\frac{2}{2}} (\sigma_1^2 \sigma_2^2)^{\frac{1}{2}}} = \frac{1}{(2\pi)^{\frac{2}{2}} |\Sigma|^{\frac{1}{2}}} \tag{4.10}$$

Exponent part can also be derived by using Exponent part of univariate Normal distribution.
From equation 4.2 we can write:

$$-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2 = -\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu) \tag{4.11}$$

Putting values in 4.3 we get:

$$= -\frac{1}{2}[x_1 - \mu_1 \quad x_2 - \mu_2] \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}^{-1} \begin{bmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \end{bmatrix} \tag{4.12}$$

and

$$\Sigma^{-1} = \frac{\begin{bmatrix} \sigma_2^2 & 0 \\ 0 & \sigma_1^2 \end{bmatrix}}{\sigma_1^2 \, \sigma_2^2} \tag{4.13}$$

Substituting Σ^{-1} in 4.12 we get

$$= -\frac{1}{2}[x_1 - \mu_1 \quad x_2 - \mu_2] \frac{1}{\sigma_1^2 \, \sigma_2^2} \begin{bmatrix} \sigma_2^2 & 0 \\ 0 & \sigma_1^2 \end{bmatrix} \begin{bmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \end{bmatrix} \tag{4.14}$$

By solving above equation we get the exponent part of Bivariate Normal distribution as:

$$= -\frac{1}{2}\left[\left(\frac{x_1 - \mu_1}{\sigma_1}\right)^2 + \left(\frac{x_2 - \mu_2}{\sigma_2}\right)^2\right] \quad (4.15)$$

where $\left(\frac{x_1-\mu_1}{\sigma_1}\right)^2 + \left(\frac{x_2-\mu_2}{\sigma_2}\right)^2$ is the equation of ellipse.

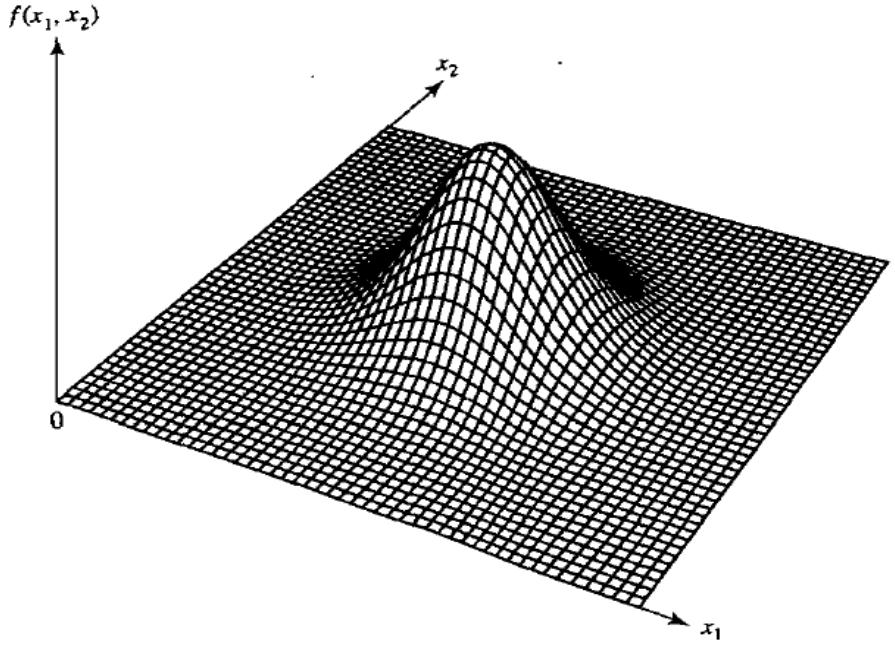


Fig 4.2 Bivariate Normal distribution $\sigma_{11} = \sigma_{22}$ and $\rho_{12} = 0$

4.2.2. Case 2: When variables are independent and $\sigma_{12} \neq 0$

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{21} & \sigma_2^2 \end{bmatrix}, \mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \text{ and } X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$$

Whereas σ_{12}^2 is equal to product of co-variance and standard deviation.

$$|\Sigma| = \sigma_1^2\,\sigma_2^2 - \sigma_{12}^2 = \sigma_1^2\sigma_2^2 - (\rho_{12}\sigma_1\sigma_2) = \sigma_1^2\sigma_2^2(1 - \rho_{12}) \quad (4.16)$$

Inverse of co-variance matrix is given by:

$$\Sigma^{-1} = \frac{1}{\sigma_1^2\sigma_2^2 - \sigma_{12}^2}\begin{bmatrix} \sigma_1^2 & -\sigma_{12} \\ -\sigma_{12} & \sigma_2^2 \end{bmatrix} \tag{4.17}$$

Substituting this value in 4.12 gives:

$$= -\frac{1}{2}[x_1 - \mu_1 \quad x_2 - \mu_2]\frac{1}{\sigma_1^2\sigma_2^2 - \sigma_{12}^2}\begin{bmatrix} \sigma_2^2 & -\sigma_{12} \\ -\sigma_{12} & \sigma_1^2 \end{bmatrix}\begin{bmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \end{bmatrix} \tag{4.18}$$

By solving and then multiply and divide with $\sigma_1^2\sigma_2^2$ we get,

$$= -\frac{\sigma_1^2\sigma_2^2}{2(\sigma_1^2\sigma_2^2 - \sigma_{12}^2)}\left[\left(\frac{x_1-\mu_1}{\sigma_1}\right)^2 - \frac{2\sigma_{12}}{\sigma_1\sigma_2}\left(\frac{x_1-\mu_1}{\sigma_1}\right)\left(\frac{x_2-\mu_2}{\sigma_2}\right) + \left(\frac{x_2-\mu_2}{\sigma_2}\right)^2\right] \tag{4.19}$$

As $\sigma_{12} = \rho_{12}\sigma_1\sigma_2$

$$= -\frac{1}{2(1-\rho_{12}^2)}\left[\left(\frac{x_1-\mu_1}{\sigma_1}\right)^2 - 2\rho_{12}\left(\frac{x_1-\mu_1}{\sigma_1}\right)\left(\frac{x_2-\mu_2}{\sigma_2}\right) + \left(\frac{x_2-\mu_2}{\sigma_2}\right)^2\right] \tag{4.20}$$

where $\left[\left(\frac{(x_1-\mu_1)^2}{\sigma_1}\right) - 2\rho_{12}\left(\frac{x_1-\mu_1}{\sigma_1}\right)\left(\frac{x_2-\mu_2}{\sigma_2}\right) + \left(x_2 - \frac{\mu_2}{\sigma_2}\right)\right]$ is the 2D equation of ellipse for Bi-variate.

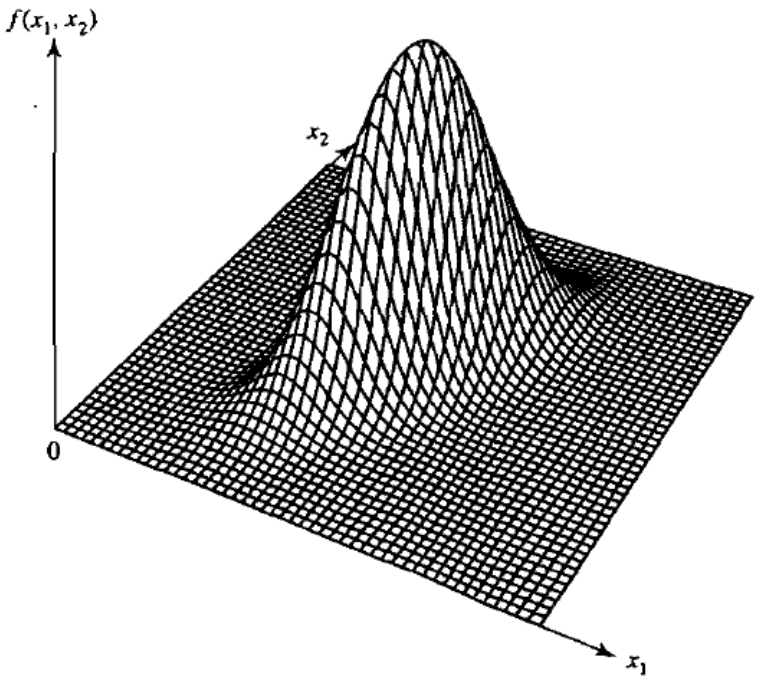


Fig 4.3 Bivariate Normal distribution $\sigma_{11} = \sigma_{22}$ and $\rho_{12} \neq 0$

4.3. Multivariate case

The multivariate Gaussian distribution for a vector **x** having D-dimensions is defined as:

$$N(x|\,\mu, \Sigma) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \; e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)} \tag{4.21}$$

Where μ is a D-dimensional mean vector, Σ is a D × D covariance matrix, which is defined as:

$$\Sigma = E[(x - \mu_i)(x - \mu_i)^T] \tag{4.22}$$

|Σ| denotes the determinant of Σ and $E[.]$ is mean value of random variable.

For $\sigma_1 = \sigma_2$

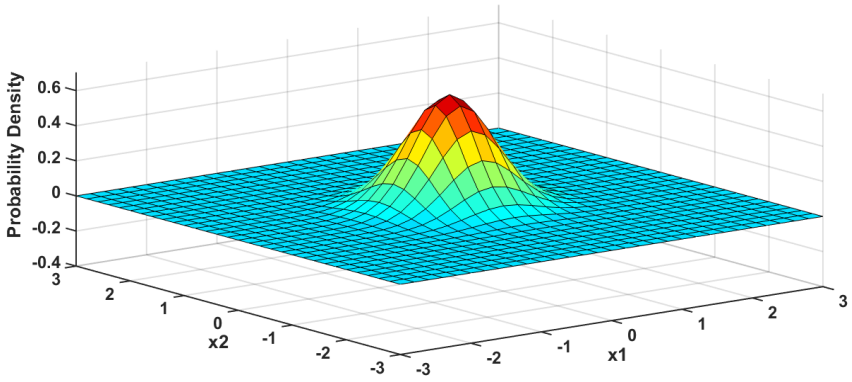


Fig 4.4 Multivariate Normal Distribution when $\sigma_1 = \sigma_2$

4.5. Properties of Multivariate Normal Distribution

- If $X_{P\times1} \sim N_P(\mu, \Sigma)$ then X_j is $N(\mu_j, \sigma^2{}_j)$ for all $X_j, j = 1,2, \dots, p$.
- If $X_{p\times1} \sim N_p(\mu, \Sigma)$ then subset of $X_{p\times1}$ $i.e$ $X_{q\times1}$ is $N_p(\mu, \Sigma)$.
- If $X_{p\times1} \sim N_p(\mu, \Sigma)$ then linear combinations of $X_j, j = 1,2, \dots, p$ is univariate normal.
- If $X_{p\times1} \sim N_p(\mu, \Sigma)$ then q linear combination of $X_j, j = 1,2, \dots, p$ is multivariate normal.

4.6. Gaussian Mixture Model (GMM)

Gaussian Mixture Model is referred to as the linear super position of Gaussian components that is usually formed by taking linear combinations of basic distribution such as Gaussian.

Normally the data in 1-d, 2-d, and 3-d is distributed in Gaussian form. The reason being its closeness to natural distribution and it is very easy to do mathematical manipulation when we have Gaussian distribution. But there may be situation when distribution is not strictly Gaussian. In such cases data sets forms clumps in their structure as shown in figure

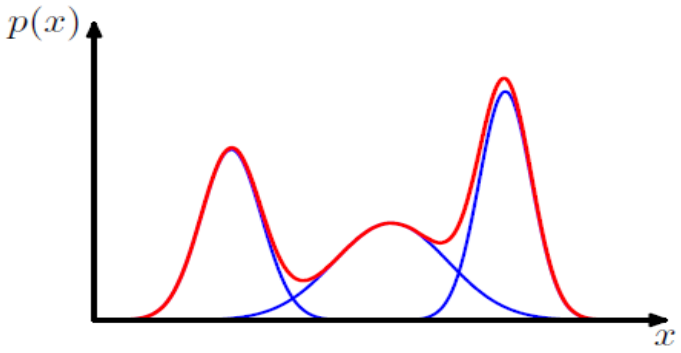


Fig-4.5 Example of a Gaussian mixture distribution in one dimension showing three Gaussians in blue and their sum in red.

It becomes very difficult to characterize such data sets under single Gaussian. Such data sets can be better characterized by linear super position of Gaussians.

By using adequate number of Gaussians and adjusting their means, covariance's as well as co-efficient in linear combination, we can approximate almost any continuous density to arbitrary accuracy. Therefore we define super position of K Gaussians densities as

$$p(x) = \sum_{k=1}^{K} \pi_k N(x|\mu_k \Sigma_k) \tag{4.23}$$

This is called mixture of Gaussians, which is shown in three dimensions as

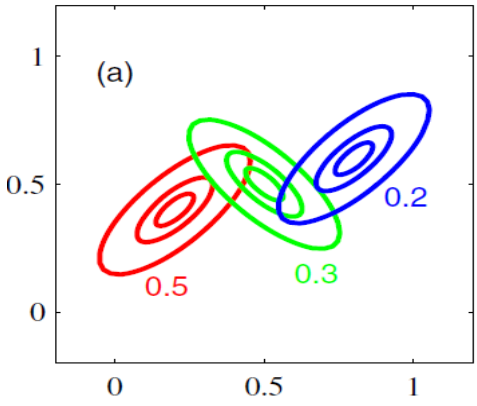


Fig-4.6 Mixture of three Gaussians

Each Gaussian density $N(x|\mu_k\Sigma_k)$ is known as component of mixture and it has its own mean and covariance. K is total number of Gaussians and the parameter π_k is known as the mixing coefficients for K^{th} Gaussian (Weightage for each Gaussian distribution). If we integrate both sides of Eq.(4.23) with respect to x and normalizing both p(x) and individual Gaussian components, we obtained

$$\sum_{k=1}^{K} \pi_k = 1 \tag{4.24}$$

Also the requirement is that $p(x) \geq 0$ along with $N(x|\mu_k\Sigma_k) \geq 0$ which implies $\pi_k \geq 0$. If we combine this condition with eq. (3.10) we get

$$0 \leq \pi_k \leq 1 \tag{4.25}$$

If we define joint probability over observed data X and latent variables, then we can obtain the distribution of observed variables by marginalization. The complex marginal distribution over observed variable can be expressed in terms of more tractable joint distribution. Hence the introduction of latent variable allows us to form complicated distribution from simpler distributions such as Gaussian Mixture.

Let's introduce a K-dimensional binary random variable z having 1 out of K representation such that for this random variable only a particular entry z_k is equal to 1, while all other elements are equal to 0. Therefore the values of z_k satisfies $z_k \in \{0,1\}$ and $\sum_k z_k = 1$. From this we can see that there are k possible states for the vector z according to which element is non-zero.

Corresponding to fig. 4.6 we will define the joint distribution $p(x,z)$ in terms of marginal distribution $p(z)$ and conditional distribution $p(x|z)$.

The marginal distribution is defined in terms of mixing co-efficient or weightage co-efficient π_k, such that

$$p(z_k = 1) = \pi_k \qquad\qquad 0 \leq \pi_k \leq 1 \tag{4.27}$$

Along with,

$$\sum_k \pi_k = 1 \tag{4.28}$$

in order to be valid probabilities. As we have defined that z uses 1 out of K representation, we can write this distribution in the form as:

$$p(z) = \prod_{k=1}^{K} {\pi_k}^{z_k} \tag{4.29}$$

In the same way, conditional distribution of X is given as:

$$p(x|z_k = 1) = N(x|\mu_k, \Sigma_k) \tag{4.30}$$

that can also be written as:

$$p(x|z_k = 1) = \prod_{k=1}^{K} N(x|\mu_k, \Sigma_k)^{z_k} \tag{4.31}$$

The joint distribution is represented by $p(z)p(x|z)$ and the marginal distribution of x can be obtained by summing the joint distribution over all possible states of z to give

$$p(x) = \sum_z p(z)p(x|z) = \sum_z \prod_{k=1}^{K} (\pi_k N(x|\mu_k, \Sigma_k))^{z_k} \tag{4.32}$$

Exploiting 1 out of K representation for z, and re-write the right hand side we have

$$\sum_{j=1}^{K} \prod_{k=1}^{K} (\pi_k N(x|\mu_k, \Sigma_k))^{I_{kj}} \tag{4.33}$$

where I_{kj} =1 if $k = j$ and 0 otherwise.

Therefore, we can write

$$\sum_{j=1}^{K} \pi_k N(x|\mu_j, \Sigma_j) \tag{4.34}$$

The marginal distribution is represented in form $p(x) = \sum_z p(x, z)$. Now if we have several observations $x_1, x_2, \dots, x_N$, it tells us that a corresponding latent variable z_n is assigned to each observed data point x_n. Hence this joint distribution instead of marginal distribution will indicate us towards significant simplifications most remarkably through the overview of Expectation Maximization algorithm.

We will also derive another important quantity that is conditional probability of z given x that is $p(z_k = 1|x)$ and $E(z_k) = p(z_k = 1|x)$ For simplicity we will denote it by $\gamma(z_k)$ and it is defined by using Bayes theorem

$$\gamma(z_k) = p(z_k = 1|x) = \frac{p(x|z)p(z)}{p(x)} \tag{4.35}$$

By putting values we get the following results

$$\gamma(z_k) = \frac{\pi_k N(x|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \, N(x|\mu_j, \Sigma_j)} \tag{4.36}$$

Where π_k is the prior probability and $\gamma(z_k)$ is the posterior probability. $\gamma(z_k)$ can also be observed as the responsibility that component k takes for explaining the observation x.

The example of 500 points that are drawn from the mixture of 3 Gaussians is shown below in Fig (4.7). These samples can be depicted from the joint distribution $p(x, z)$ by plotting data points at the equivalent values of x and then allot them colors according to z value.

Similarly, if we ignore the values of z shown in (b), we can obtain samples for marginal distribution p(x) by the joint distribution.

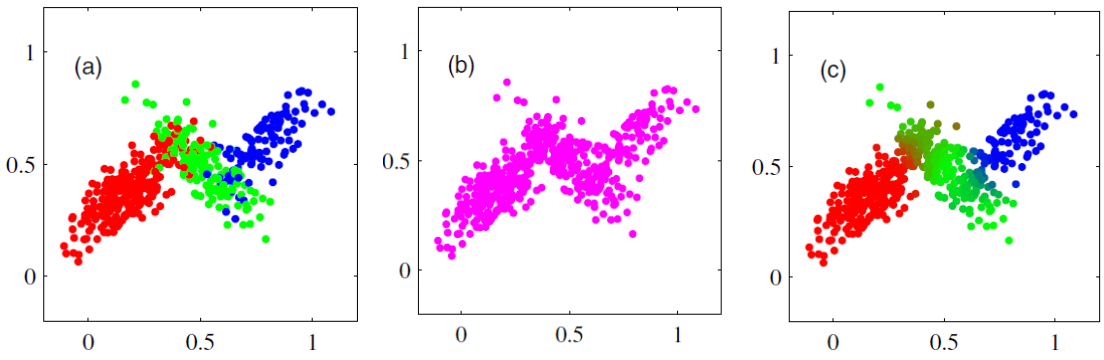


Fig 4.7 (a) 500 points drawn from Gaussian mixture by the joint distribution p(z)p(x|z). The three states of z represented by 3 Gaussians are depicted in red, green, and blue (b) Samples obtained by not considering the values of z and just plotting the values of x (c) The samples representing the value of the responsibilities $\gamma(z_{nk})$ for each color linked with data point x_N.

This synthetic data in (b) can also be used to illustrate the responsibility by estimating the posterior probability for each component for every data. If x_N is drawn from k^{th} component, $z_{nk} = 1$ while all others are 0. We can represent the values of responsibilities $\gamma(z_{nk})$ associated with data point x_N by using proportions of red, blue and green colors that is given by $\gamma(z_{nk})$ for k=1,2,3 respectively. Here, the point having $\gamma(z_{n1}) = 1$ will be colored as red whereas the one having $\gamma(z_{n2}) = \gamma(z_{n3}) = 0.5$ will be colored with equal quantities of blue and green, hence appeared as cyan.

4.6.1 Maximum Likelihood Estimation

One method to estimate the parameters μ and Σ is Maximum likelihood Estimation (MLE). If we have a data set of N observations $\{x_1, x_2, \dots, x_N\}$ and we want to model this data by utilizing the mixture of Gaussians.

If we represent this data set as $N \times D$ matrix X and corresponding latent variables as $N \times K$ matrix Z and assumed that data points are drawn from the distribution independently, then graphical representation of this data will be through Gaussian mixture model. From eq. (3.9), the log likelihood function is given by

$$\ln p(X|\mu,\pi,\Sigma) = \sum_{n=1}^{N} \ln\left\{\sum_{k=1}^{K} \pi_k N(x_n|\mu_k,\Sigma_k)\right\} \tag{4.37}$$

Maximizing this function turns out to be more complex problem because of summation over k inside logarithm.

By setting the derivatives of log likelihood to zero, we will not get a closed form solution. Hence we will use another technique which is known as Expectation Maximization EM.

4.6.2 Expectation Maximization Algorithm

Expectation maximization is a dominant method that is used for finding maximum likelihood solutions for model having latent variables.

From 3.13, we have

$$\ln p(X|\mu,\pi,\Sigma) = \sum_{n=1}^{N} \ln\left\{\sum_{k=1}^{K} \pi_k N(x_n|\mu_k,\Sigma_k)\right\} \tag{4.38}$$

By taking derivative of above eq. with respect to mean μ_k and equate it to zero we have,

$$\frac{\partial}{\partial \mu_k}\left(\sum_{n=1}^{N} \ln\left\{\sum_{k=1}^{K} \pi_k N(x_n|\mu_k,\Sigma_k)\right\}\right) = 0 \tag{4.39}$$

$$\sum_{n=1}^{N} \frac{\pi_k \ \frac{\partial}{\partial \mu_k} N(x_n|\mu_k,\Sigma_k)}{\sum_{l=1}^{K} \pi_l N(x_n|\mu_l,\Sigma_l)} = 0 \tag{4.40}$$

By simplifying the above eq. we get,

$$\sum_{n=1}^{N} \frac{\pi_k}{\sum_{l=1}^{K} \pi_l N(x_n|\mu_l,\Sigma_l)} \times \ 1/2\pi^{\frac{2}{2}}\,\Sigma^{\frac{1}{2}} \times e^{-\frac{1}{2(x-\mu)\Sigma^{-1}(x-\mu)^T}}\left(-\frac{2}{2}\right)(x_n-\mu_k)(-1) = 0 \tag{4.41}$$

and

$$\sum_{n=1}^{N} \gamma(z_{nk}) \times \frac{1}{\Sigma_k} \times (x_n - \mu_k) = 0 \tag{4.42}$$

$$\sum_{n=1}^{N} \gamma(z_{nk}) x_n = \sum_{n=1}^{N} (\mu_k) \tag{4.43}$$

This gives the value of mean

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n \tag{4.44}$$

By setting the derivatives $\ln p(X|\mu, \pi, \Sigma)$ with respect to Σ_k to zero and follow the same line of reasoning, we obtain the following result,

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \, (x_n - \mu_k)(x_n - \mu_k)^T \tag{4.45}$$

At the end, we will maximize $\ln p(X|\mu, \pi, \Sigma)$ with respect to π_k that is mixing coefficient and taking into account constraint 3.12, which requires mixing co-efficient to some to 1. It can be accomplished by using a Lagrange multiplier and maximizing the following quantity and equate it to zero

$$\ln p(X|\mu, \pi, \Sigma) + \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right) \tag{4.46}$$

By simplifying,

$$0 = \sum_{n=1}^{N} \frac{N(x_n|\mu_k, \Sigma_k)}{\sum_{l=1}^{K} \pi_l N(x_n|\mu_l, \Sigma_l)} + \lambda \tag{4.47}$$

Multiplying both sides by π_k we obtain:

$$0 = \sum_{n=1}^{N} \gamma(z_{nk}) + \lambda \pi_k \tag{4.48}$$

Which gives

$$0 = N_k - N\pi_k \tag{4.49}$$

Hence,

$$\pi_k = \frac{N_k}{N} \tag{4.50}$$

3.6.2.1 EM algorithm for GMM

Given a Gaussian mixture model, goal is to maximize the likelihood function with respect to the parameters comprising means and covariances of components and mixing coefficients.

Step-1

Initialize the means μ_j, covariance Σ_j and mixing coefficient π_j. Then evaluate the initial value of the log likelihood where $j = 1,2,3, \ldots, k$.

Step-2

E-Step: Evaluate responsibilities using current parameter values.

$$Y_k(x) = \frac{\pi_k N(x|\mu_k \Sigma_k)}{\sum_{j=1}^{K} \pi_j \, N(x|\mu_j, \Sigma_j)} \tag{4.51}$$

where $Y_k(x)$ is latent variable for k^{th} Gaussian.

Step-3

M-step: Re-calculate the parameters by using current obtained values

$$\mu_j = \frac{\sum_{n=1}^{N} Y_{j\,(x_n)}\, x_n}{\sum_{n=1}^{N} Y_j\,(x_n)} \tag{4.52}$$

$$\Sigma_j = \frac{\sum_{n=1}^{N} Y_j(x_n)\,(x_n-\mu_j)(x_n-\mu_j)'}{\sum_{n=1}^{N} Y_j(x_n)} \tag{4.53}$$

$$\pi_j = \frac{1}{N}\sum_{n=1}^{N} Y_j(x_n) \tag{4.54}$$

Step-4

Evaluate log-likelihood,

$$\ln p(X|\mu,\Sigma,\pi) = \sum_{n=1}^{N} \ln\left\{\sum_{k=1}^{K} \pi_k N(x_n|\mu_k,\Sigma_k)\right\} \tag{4.55}$$

We have to find out that if these parameters truly represent data points. If this does not happen, go back to step-2, which will help to converge to better solution.

CHAPTER 5

Hidden Markov Model

To model the process in a way that we have a state for each of the observation on the data is a very powerful assumption if the real world applications are considered. Computing the probability distribution for each and every state is not always feasible in modeling such data. This is because it is quite impractical approach to estimate all the transition probabilities for the amount of required data set.

Markov chains find its extension to a well-known process that is used practically known as Hidden Markov Models (HMMs). These models offer the solution by introducing the hidden states (that are not observed) and each state can be easily estimated by generated observations from the given data set.

In simpler Markov models (like a Markov chain) the states are known to observer, and therefore state transition probabilities are the only parameters. In Hidden Markov Model, state is not directly known but output is visible and it depends on state. Probability distribution of each state is present over the possible output tokens. Therefore from the sequence of these tokens that ae generated by HMM, we can get some information about the sequence of states. It is important to note at this point that the states we are referring to as hidden states are not the parameters of the model but they are the states of the Markov Chain when Hidden Markov Model is defined.

Hidden Markov models are especially known for their application such as statistical signal Analysis 1-D, Image Modeling 2-D, handwriting, gesture recognition, speech, part-of-speech tagging, etc.

HMMs have a historical background mainly consisting of two parts. One has its relevance with Markov processes and Chains while the other part it deals with the algorithms necessary to define a Hidden Markov Model for solving problems in the field of computer sciences and other domains to serve the needs for modern sciences.

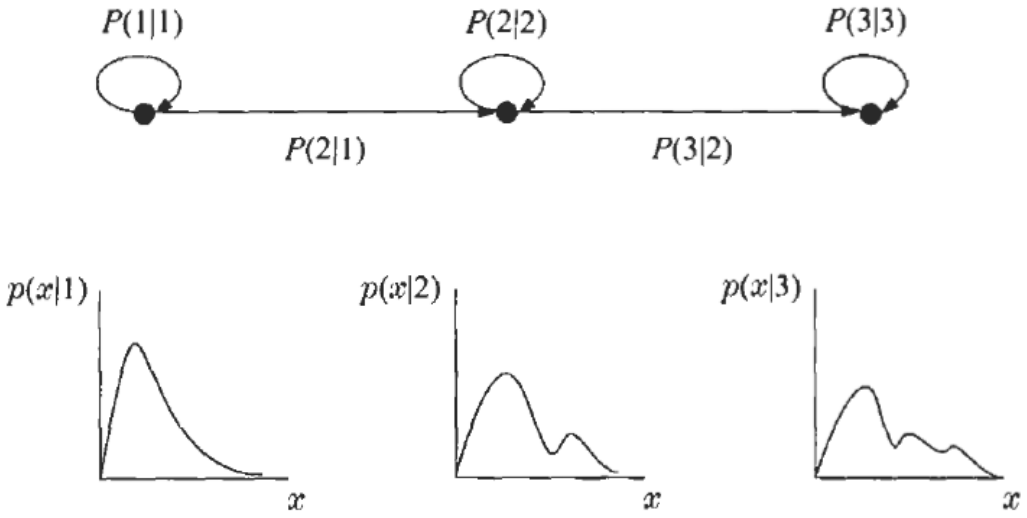


Fig 5.1 Model parameters describing a three-state hidden Markov model.

5.1. From observable to hidden state

HMMs provide great help when there is need of modeling a process in which we do not have direct knowledge about the present state of system. The only direct knowledge that we have about the process to model is the set of observations it generates while we don't have direct access to the internal structure of process. It is easier to build a model that gives good approximation of process, when we have specific knowledge of the domain, but, in many cases this problem doesn't have easy solution and it can be task-dependent.

5.2. Parameters of HMM

For Markov chains, the output symbols and the states are same. In other words, the observation is the same as the state. But in part of speech tagging, we have output symbol or observation known as 'words' and the hidden states are the part of speech tags.

In Hidden Markov Model we don't know in which state we are in. The following equation is guaranteed to give us the best tag sequence.

$$\hat{t}_n^1 = argmax_{t_n^1} P(t_n^1|w_n^1) \quad (5.1)$$

Where t_n^1 and w_n^1 represents the speech tag sequence and word sequence from $1, \dots, n$ respectively. To make it operational, Bayes rule is applied to transform this equation into set of other probabilities.

$$\hat{t}_n^1 = argmax_{t_n^1} \frac{P\left(w_n^1 | t_n^1\right) P(t_n^1)}{P(w_n^1)} \tag{5.2}$$

Length of observation sequence	L
Number of states	S
Number of observations	O
States	$Q = (q_1, q_2, \dots, q_n)$
Possible symbols	$V = (v_1, v_2, \dots, v_m)$
Transition Probability Matrix	$A = \{a_{ij}\}$
Output Probability Matrix	$B = \{b_j(O_k)\}$
Initial State Vector	π

Table 5-1 Parameters of Hidden Markov Model

5.3. A motivating Example

For any non-trivial task whenever we are asked to do a task, we find that information that we have to work with is very much partial. In such cases we have to deal with uncertain information. Let's suppose that we have three urns and each of them contain Red, Green and Blue balls as shown in fig.

Again suppose that person is picking ball from these urns and it gives us a pattern of drawing a ball from these urns as RRGGBRGR.

We have to find out the sequence of urns from which he had drawn the balls. Hence, for this sequence of colors of balls, we have to produce urns sequence or state sequence that is given as,

$$\{U_1, U_2, U_3, \dots, U_i\} \tag{5.3}$$

We have information of probability of transition from one urn to any other urn. For example if a person draw a ball from urn 1 and then drawing a next ball from urn 1 again is 0.1. Table shows the probabilities of having balls of specific color in an urn. Since we know the number of balls and their colors, and we are also given the probabilities of these colors. These two things are known to us and using these, we have to compute the most probable state sequence that is hidden to us. We are given

Observation Sequence= RRGGBRGR

# of Red = 30	# of Red = 10	# of Red = 60
# of Green = 50	# of Green = 40	# of Green = 10
# of Blue = 20	# of Blue = 50	# of Blue = 30

Fig.5.2 Three Urns containing Red, Green and Blue balls

	U1	U2	U3
U1	0.1	0.4	0.5
U2	0.6	0.2	0.2
U3	0.3	0.4	0.3

Table 5-2 (a) Probability of transition

	R	G	B
U1	0.3	0.5	0.2
U2	0.1	0.4	0.5
U3	0.6	0.1	0.3

Table 5-2 (b) Probability of drawing a ball

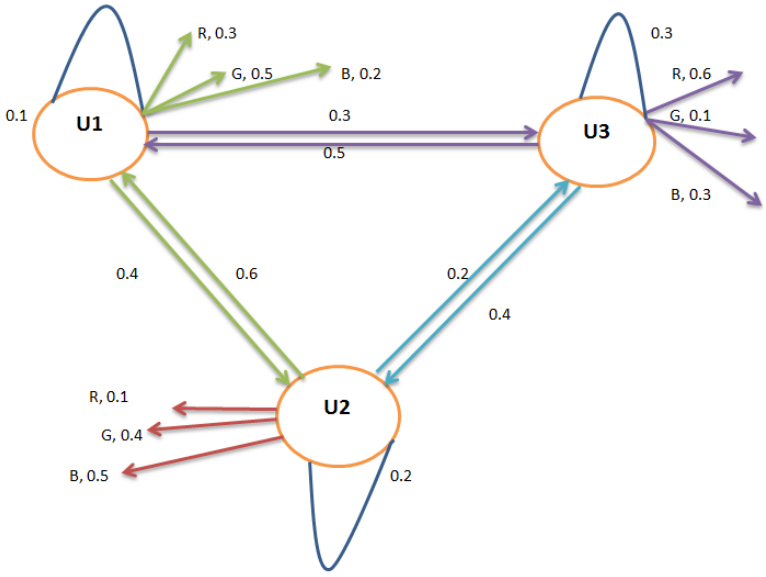


Figure 5.3: Diagrammatic Representation

Hence, based on nations used, the data can be summed up as follows

$$S = \{U1, U2, U3\}$$

$$V = \{R, G, B\}$$

$$O = \{o_1, o_2, \dots, o_n\}$$

$$Q = \{q_1, \dots, q_n\}$$

$\pi_i = P(q_1 = U_1)$ (Initial probability that the systems starts in the initial state U1, usually this is taken)

	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8
Observations	R	R	G	G	B	R	G	R
States	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8

Table 5-3 Observations and States

The above table shows the pattern of drawing balls as RRGGBRGR with observations $O_1, O_2, \dots, O_8$ and set of states $S_1, S_2, \dots, S_8$. From the given table we can see that we have to find out the most probable state sequence that is hidden to us. Hence our goal is to maximize *P(S | O)*, refer back to equation 4.1 that can be re-written as:

$$S^* = argmax_s\big(P(S|O)\big) \qquad (5.4)$$

This process leads to efficient computation because of Markov assumption that takes those leaves of the tree that have highest probability. Markov assumption states that the probability of

a state being the state of the machine depends only on the previous state (Order 1 Markov assumption). Applying Markov assumption to *P(S | O)* leads to;

$$P(S|O) = P(S_1|O)P(S_2|S_1, O)P(S_3|S_2, O), \dots, P(S_8|S_7, O) \quad (5.5)$$

These probability terms are not easily solvable as there are other cumbersome items that need to be processed for it. Bayes' theorem along with Markov assumption serves the purpose in this regard that is discussed in the next section.

5.4 Essentials of Hidden Markov Model

- **Markov Assumption + Naïve Bayes**

  Bayes theorem invoked along with Markov assumption is a powerful tool for problem solving used in statistical artificial intelligence and machine learning. Applying this theorem, the State transition and observation sequence probabilities will be discussed below. This is discussed in reference to the above example.

- **State Transitions Probability**

  The prior *P(S)* can be treated in the following way

$$P(S) = P\ (S_{1-8}) \quad (5.6)$$

$$P(S) = P(S_1)\ P(S_2|S_1)\ P(S_3|S_{1-2}) \dots \dots\ P(S_8|S_{1-7}) \quad (5.7)$$

  By Markov assumption, (k=1)

$$P(S) = P(S_1)P(S_2|S_1)P(S_3|S_2) \dots \dots P(S_8|S_7) \quad (5.8)$$

- **Observation Sequence Probability**

  The next probability component is *P (O | S)* which is found out to be:

$$P(O|S) = P(O_1|S_{1-8})\ P(O_2|O_1, S_{1-8})\ P(O_3|O_{1-2}, S_{1-8}) \dots\dots\ P(O_8|O_{1-7}, S_{1-8}) \quad (5.9)$$

Assumption that the ball drawn depends only on the urn chosen

$$P(O|S) = P(O_1 \mid S_1)\ P(O_2 \mid S_2) P(O_3 \mid S_3) \dots\dots P(O_8 \mid S_8) \quad (5.10)$$

By applying Bayes' theorem we have,

$$argmax_s P(S|O) = argmax_s P(S)\ P(O|S) \quad (5.11)$$

Here the denominator $p(O)$ is ignored because it is independent of S so it can be eliminated from consideration. Hence by putting values in 4.6 we have

$$P(S|O) = P(S_1)\ P(S_2|S_1)\ P(S_3|S_2) \dots\dots P(S_8|S_7)\ P(O_1|S_1)\ P(O_2|S_2)$$
$$P(O_3|S_3) \dots\dots P(O_8|S_8) \quad (5.12)$$

All these terms can be grouped together in the way shown in an equation below

$$P(O_k|S_k).P(S_{k+1}|S_k) = P\left(S_k \overset{O_k}{\rightarrow} S_{k+1}\right) \quad (5.13)$$

The diagram shown below shows set of observations with corresponding states.

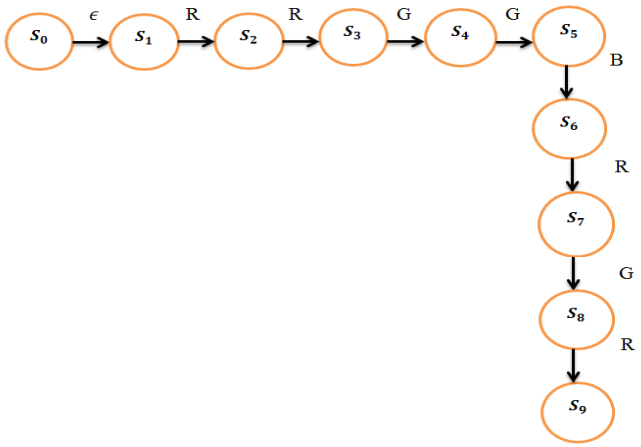


Fig 5.4: Diagrammatic Representation of Observations and State

5.5 Properties of Hidden Markov Model

There are two main properties of Markov processes on which the theory of HMM rests.

- **Limited Horizon**

It states that given previous ***t*** states, a state ***i*** is independent of the preceding 0 to $t - k + 1$ states. This means that beyond *k* states which come before the t_{th} state, everything can be ignored. This is the limited horizon or the window property and the process is known as k Markov process.

$$P(X_t = i \mid X_{t-1}, X_{t-2}, \dots, X_0) = P(X_t = i \mid X_{t-1}, X_{t-2}, \dots, X_{t-k}) \qquad (5.14)$$

- **Time Invariance**

The dependence of one particular state on the previous state is observed over the whole sequence in Markov process. This means that the conditional probability shown in the equation below is position invariant that is it does not change from place to place in a sequence.

$$P(X_t = i \mid X_{t-1} = j) = P(X_1 = i \mid X_0 = j) = P(X_n = i \mid X_{n-1} = j) \qquad (5.15)$$

5.6 Probability Laws

In machine learning and natural language processing, there are two essential probability laws.

- **Chain Rule**

Chain rule helps to break up the complete sequence into set of terms

$$P(X_1, X_2, \dots, X_k) = P(X_1)\, P(X_2|X_1)\, P(X_3|X_2X_1)\, P(X_k|X_{k-1}X_{k-2} \dots X_1) \qquad (5.16)$$

- **Marginalization**

$$P(A) = \Sigma\, P(A, B_1, B_2, B_3 \ldots B_n) \tag{5.17}$$

Where $(B_1, B_2, B_3 \ldots B_n)$ takes all the possible values

These two rules are a Hidden Markov Model Probability law that is extremely important in Maximum Likelihood, Natural Language Process and we have to make use of them in different times.

5.7 Three problems in Hidden Markov Model

Three of the basic problems related to HMMs are described briefly in this section. Also, the solutions to these problems are also given with efficient algorithms.

5.7.1 Evaluation Problem

Consider the model that is given and it is represented by $\theta = (A, B, \pi)$ and a sequence of observations denoted by *O*, we have to compute the probability that a particular output sequence was produced by that model θ.

5.7.2 Decoding Problem

For a given model, $\theta = (A, B, \pi)$ and an observation sequence *O*, we would like to find most probable sequence of Hidden states that has led to generation of the given set of observations. In other words, it means that we want to determine the hidden parts that are contained in the Hidden Markov Model.

5.7.3 Learning Problem

Given a set of output sequences that is set of visible states O and set of hidden states S, we have to find out the set of transition probabilities a_{ij} and $b_j(O_k)$.

5.8 Solution to Problems

Each of the problems mentioned above has its own solution which is discussed in the following sections.

5.8.1 Solution to Problem 1: Forward & Backward Probability Algorithm

5.8.1.1 Forward Probability Algorithm

The forward probability $F(k, i)$ is defined as probability of being in a state S_i having seen observations $O_0, O_1, O_2, \dots, O_k$. M being the length of sequence

$$F(k, i) = P(O_0, O_1, O_2, \dots, O_k, S_i) \tag{5.18}$$

The probability of observed sequence P (observed) is $P(O_0, O_1, O_2, \dots, O_m)$ which is marginalized to obtain following equation with *N* being the number of states

$$P(O_0, O_1, O_2, \dots, O_m) = \sum_{i=0}^{N} P(O_0, O_1, O_2, \dots, O_m, S_p) \tag{5.19}$$

Hence, the final forward probability comes out to be;

$$P(O_0, O_1, O_2, \dots, O_m) = \sum_{i=0}^{N} F(m, p) \tag{5.20}$$

Now, the question arises how to efficiently get this $F\ (m, p)$? For this we have to peep into the sequences $(O_0, O_1, O_2, \dots, O_k)$. We have the states and observations as shown in the figure below. Starting form S_0 it can go to any state and state transition is $S_p \rightarrow S_q$ on symbol O_k.

	O_0	O_1	O_2	….	O_{k-1}	O_k	O_{k+1}	…. O_m
$S_{0\rightarrow}$	S_1	S_2	….		$S_p \rightarrow S_q$		S_m	

Hence, the forward probability is summed up as;

$$F(k,q) = P\big(O_0, O_1, O_2, \dots, O_k, S_q\big) \tag{5.21}$$

$$F(k,q) = P\big(O_0, O_1, O_2, \dots, O_{k-1}, O_k, S_q\big) \tag{5.22}$$

By Marginalization,

$$F(k,q) = \sum_{p=0}^{N} P(O_0, O_1, O_2, \dots, O_{k-1}, O_k, S_q) \tag{5.23}$$

Applying chain rule,

$$F(k,q) = \textstyle\sum_{p=0}^{N} P\big(O_0, O_1, O_2, \dots, O_{k-1}, S_p\big)\, \mathrm{P}(O_k, S_q | \big(O_0, O_1, O_2, \dots, O_{k-1}, S_p\,\big) \tag{5.24}$$

$$F(k.q) = \sum_{p=0}^{N} F(k-1,p) P(O_k, S_q | S_p) \tag{5.25}$$

$$F(k.q) = \textstyle\sum_{p=0}^{N} F(k-1,p)\; P(S_p \overset{O_k}{\rightarrow} S_q) \tag{5.26}$$

$$F(k,q) = \sum_{p=0}^{N} F(k-1,p) \tag{5.27}$$

This equation 4.14 is the recursive expression for Forward Probability giving us a recursive algorithm to compute the forward probability. Complexity of forward probability calculation is nothing but Length of states multiplied by the length of observed sequence is |S|. |O|. The expression for computing $F\ (k, q)$ is;

$$T_k = \sum_{i=0}^{N} T_{k-1} \tag{5.28}$$

5.8.1.1.1 Boundary Conditions for Forward Algorithm

The boundary condition for Forward Algorithm is

$$F(0, q) = P_q \tag{5.29}$$

where P_q is the initial probability of being in state S_q that is $S_p \rightarrow S_q$. The forward probability can be computed very easily and it is not difficult to show that it can be computed in time proportional to length of observation sequence. So it is a linear time computation.

5.8.1.2 Backward Probability Algorithm

Backward probability $B\ (k, i)$ is defined as seeing the symbols $O_k, O_{k+1}, O_{k+2}, \dots, O_m$ given the state S_i. *M* being the length of sequence

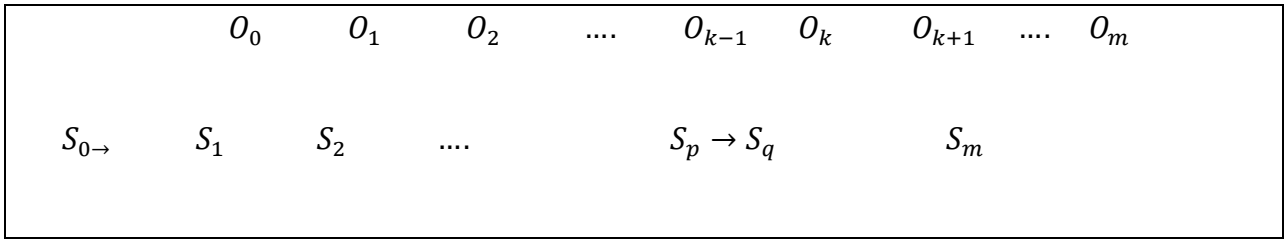
$$B(k, i) = P(O_k, O_{k+1}, O_{k+2}, \dots, O_m | S_i) \tag{5.30}$$

The probability of observed sequence P (observed) is $P(O_0, O_1, \dots, O_m)$ which is marginalized to obtain following equation with *N* being the number of states. The probability of the observed sequence is a single backward probability with the argument as 0,

$$P(O_0, O_1, \dots, O_m) = \sum_{i=0}^{N} P(O_0, O_1, \dots, O_m \mid S_0) \tag{5.31}$$

$$P(O_0, O_1, \dots, O_m) = B(0,0) \tag{5.32}$$

The backward probability is calculated same as that of forward probability. Referring to the diagram again, where we have state transition $S_p \rightarrow S_q$ over the symbol O_k.Backward probability can be expressed as follows



$$B(k,p) = P(O_k, O_{k+1}, O_{k+2} \ \dots, O_m \mid S_p) \tag{5.33}$$

$$B(k,p) = P(O_k, O_{k+1}, O_{k+2} \ \dots, O_m, O_k \mid S_p) \tag{5.34}$$

$$B(k,p) = \sum_{q=0}^{N} P\big(O_k, O_{k+1}, O_{k+2} \ \dots, O_m, O_k, S_q \big| \ S_p\big) \tag{5.35}$$

$$B(k,p) = \sum_{q=0}^{N} P\left(O_k, S_q \middle| S_p\right) P(O_k, O_{k+1}, O_{k+2}, \dots, O_m | O_k, S_q, S_p) \tag{5.36}$$

$$B(k,p) = \sum_{q=0}^{N} P(O_k, O_{k+1}, O_{k+2} \ \dots, O_m \mid S_q) \,.\, P\big(O_k, S_q \big| S_p\big) \tag{5.37}$$

$$B(k,p) = \sum_{q=0}^{N} B(k+1, q) \,.\, P\left(S_p \overset{O_k}{\rightarrow} S_q\right) \tag{5.38}$$

Backward probability is also a linear time computation as that of forward probability. For any observed sequence and the corresponding state sequence, we have the notion for the k_{th} place and for any point in the stream we can compute the forward probability up to any point and backward probability from that point to the end of the observation sequence.

5.8.1.2.1 Boundary Conditions for Backward Algorithm

We have seen the expression for backward algorithm (equation4.16). The term (k+1) in the equation goes on increasing till the end of the observation sequence. So we must have a boundary condition for this algorithm.

Having observed the last symbol in the whole observation sequence, the system is said to be in the final state. So, the transition from S_m to S_{final} with the output symbol as O_m is the boundary condition for backward algorithm $(S_m \xrightarrow{O_m} S_{final})$. Hence $B(k,p)$ is obtained from the last symbol; where S_{final} is one of the states of Hidden Markov Model.

5.8.2 Solution to Problem 2: Viterbi Algorithm

The decoding problem of hidden markov model, which seeks to find the best (or optimal) state sequence associated with a given observation sequence O of a given model λ can be solved recursively by using Viterbi algorithm.

The Viterbi algorithm is a dynamic programming algorithm used to find the most likely sequence of hidden states known as Viterbi path that gives us a sequence of observed events, especially in the framework of Markov information sources and Hidden Markov models. It is now also commonly used in speech recognition, keyword spotting, computational linguistics, speech synthesis, and bioinformatics.

Consider the example discussed above in Hidden Markov Model section, where we have three urns denoted by S1, S2 and S3.We have to find the state sequence when the observation sequence is given.

The Pattern of drawing a ball from these urns is given as RRGGBRGR. State sequence can be found using Viterbi algorithm. For this we have to calculate probabilities to find out shortest path.

Where initial probability vector is [1 0 0].

Step 1: Our starting state is S1.So probability of S1 is 1, for S2 and S3 probabilities are zero.

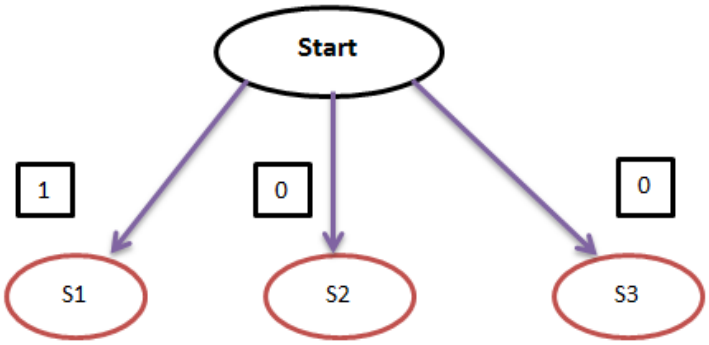
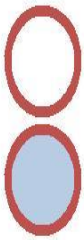


Fig 5.5: Initial Tree Diagram for Viterbi algorithm example

While considering red, blue or green ball, states probabilities can be calculated generally as:

Transition probabilities*Emission probabilities*previous probabilities

Initially we will extend S1 and then we will further extend the nodes S1, S2 and S3 and then take the highest probabilities of S1, S2 and S3 for further extension of tree. This is the way the tree will grow further. Complete Tree diagram using Viterbi algorithm is given below along with probability for every state for given observation on below or side of that state.



Red circle denoted maximum value chosen for states S1, S2 and S3 for given observation. Obtained by process of back tracking are our required most likely path for above observations.

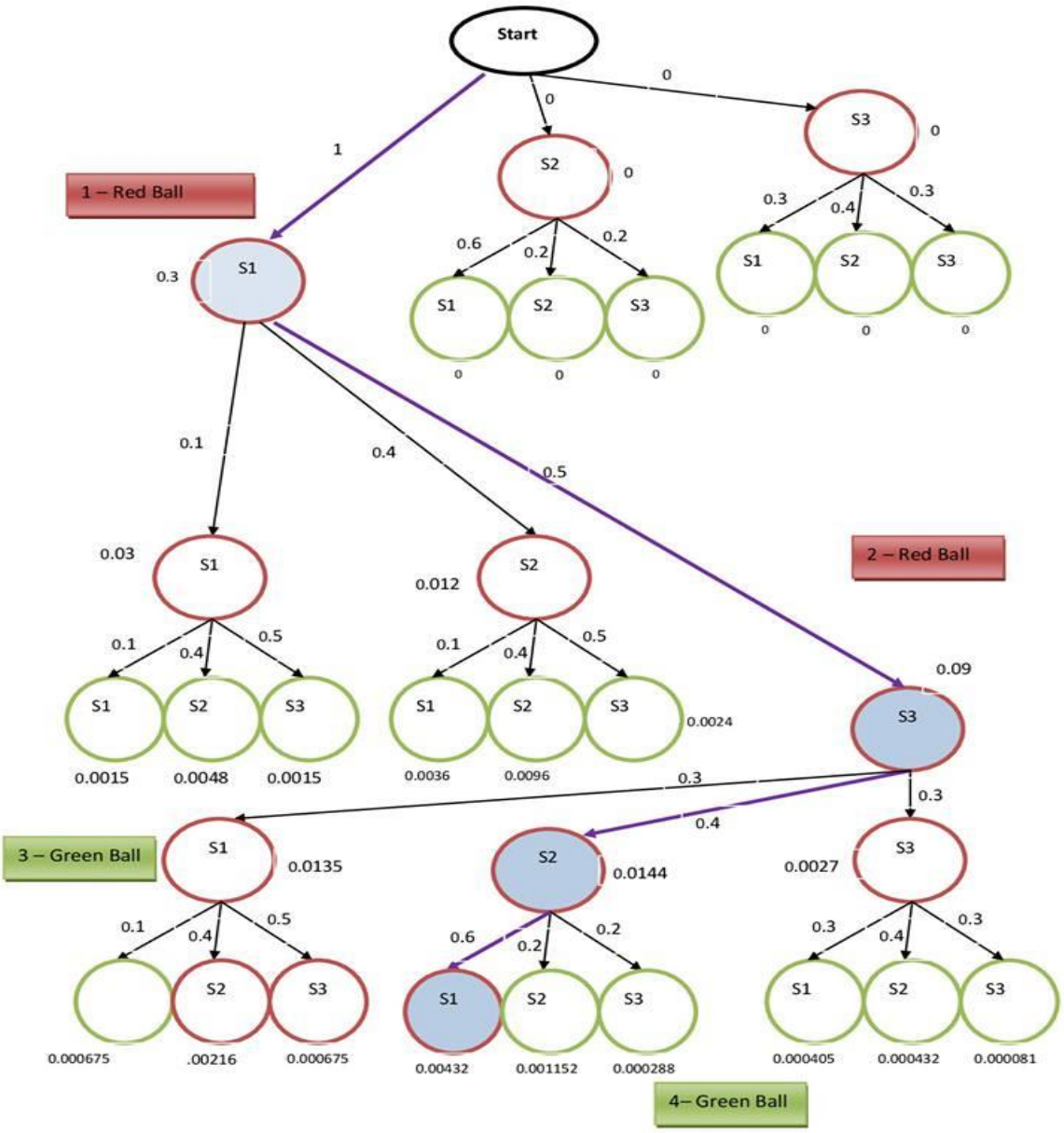


Fig 5.6 (a): Tree Diagram for Viterbi algorithm example

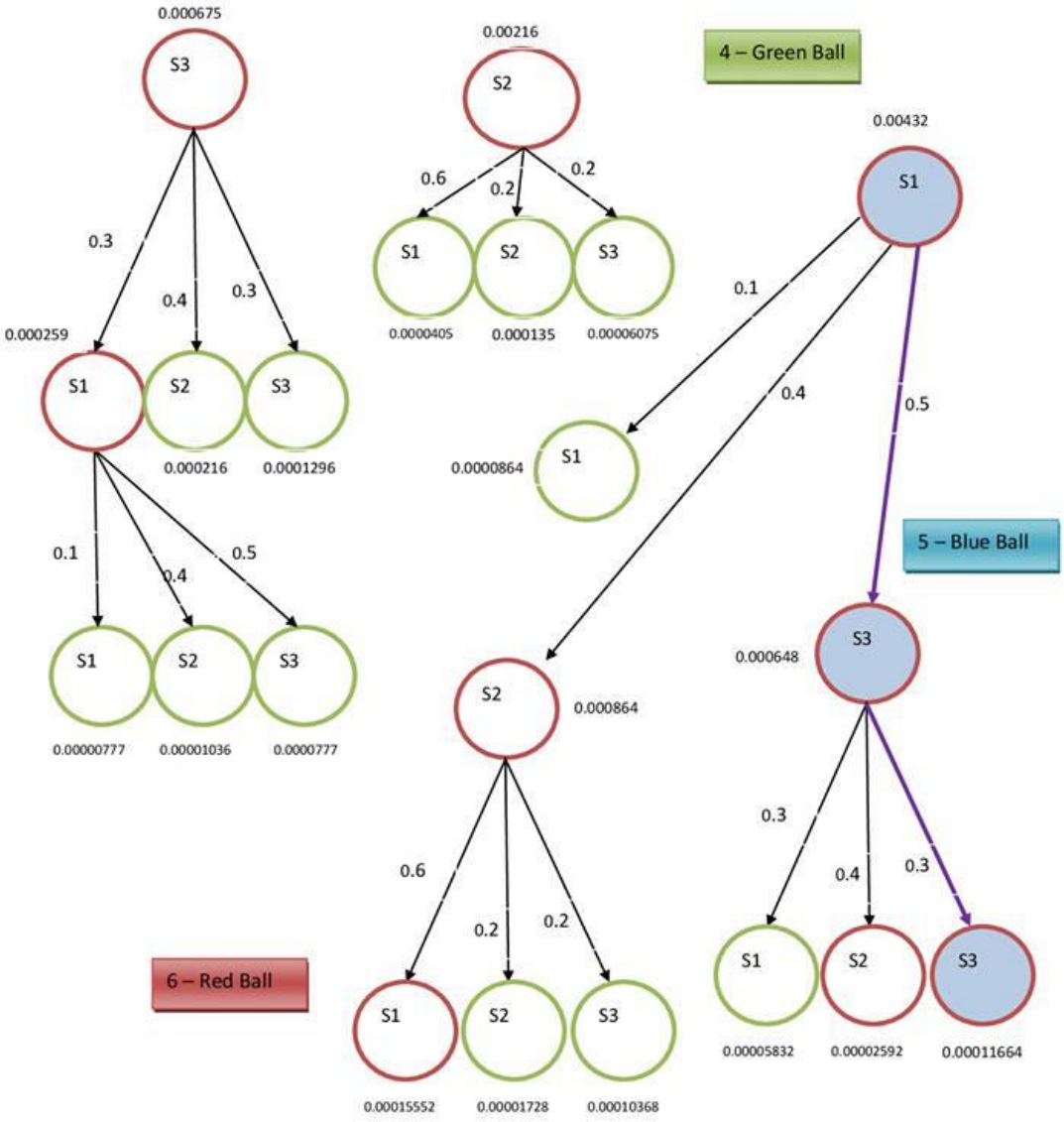


Fig 5.6 (b): Tree Diagram for Viterbi algorithm example

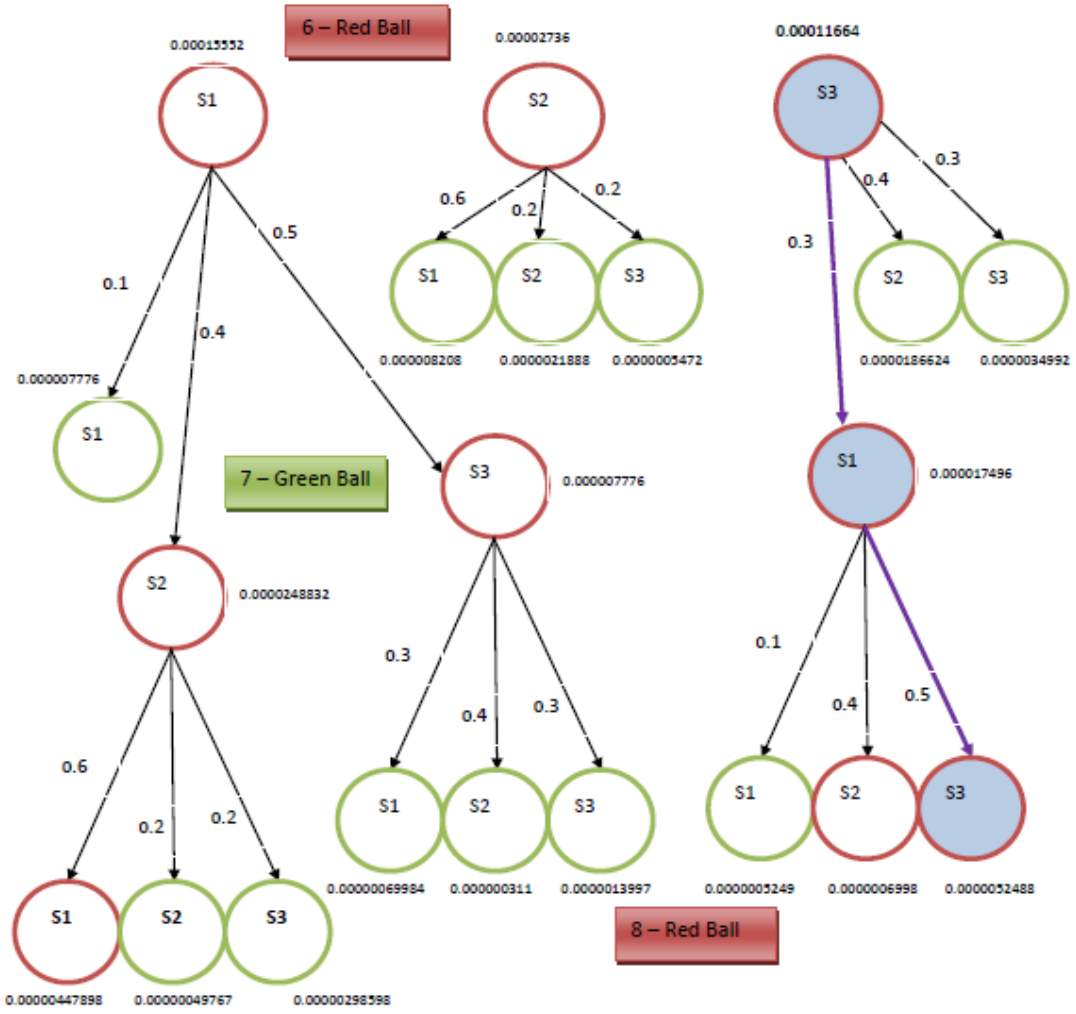


Fig 5.6 (c): Tree Diagram for Viterbi algorithm example

So in the last step maximum value among values of S1, S2 and S3 is of S3. So the maximum probability for the observations **O = R R G G B R G R** is that of S3 in last step.

So probability of most likely states which produces this observation is maximum value of S3 in the last step which is **P* = .0000052488**.

Now by process of backtracking from S3 we get the states that are most probable for above observations, also highlighted in blue circles, as **S1 S3 S2 S1 S3 S3 S1 S3**.

5.8.2.1 Steps in Viterbi Algorithm:

Given:

1. Hidden Markov Model
   - Initial state: S_1
   - Alphabet: A=$\{a_1, a_2, \dots, a_p\}$
   - Set of States: $S = \{S_1, S_2, \dots, S_n\}$
   - Transition Probability $P(S_i \xrightarrow{a} S_j)$
2. The Output String $\{a_1, a_2, \dots, a_T\}$

To Find:

The most likely sequence of states $C_1, C_2, \dots, C_T$ which produces the given output sequence. The Data Structure for this algorithm is defined as;

Data Structure

- An N*T array called SEQSCORE to maintain the winner sequence always (N= number of states, T= Length of the output sequence)
- Another N*T array called BACKPTR to recover the path.

Steps:

Three basic steps in Viterbi algorithm are:

- Initialization
- Iteration
- Sequence Identification

I. Initialization:

SEQSCORE (1, 1) = 1.0

BACKPTR (1, 1) = 0

For (i=2 to N) do

SEQSCORE (i, 1) = 0.0

[Expressing the fact that first state is S1]

Step 1 shows that S1 is the starting state and in step 2 we have that there is no state before this. In step 3 we defined that we make probability value 0 in all other states except S1.

II. Iteration

For (t=2 to T) do

For (i=1 to N) do

$SEQSCORE(i,t) = Max_{j=1,N}$

$\left[SEQSCORE\left(j,(t-1)\right).P\left(S_i \overset{a_k}{\rightarrow} S_j\right)\right]$

$BACKPTR\ (I,t) = index\ j\ that\ gives\ the\ maximum\ above$

In first step we go of our observation sequence symbol by symbol where T is the length of observation sequence and for every symbol on the observation sequence. In second step we do iteration over the number of states so this is to record the state in which sequence is ending. In third step we have to make sure that we only advance k states at particular level, we do not advance any state whose probability value is less than the winner sequence probability value ending in particular state.

Fourth step shows the multiplication of accumulated sequence probability by the transition probability. Last step is a way of keeping the pointer to be able to recover the state sequence.

III. Sequence Identification

$C(T)$= i that maximizes SEQSCORE (i, T)

For i from (T-1) to 1 do

$$C(i) = BACKPTR\ [C(i+1),(i+1)]$$

It shows the state sequence which has found to be the highest probability state sequence.

5.8.3 Solution to Problem 3: Baum- Welch (Forward-Backward Algorithm)

Baum Welch algorithm is the synthesis of both the Forward Algorithm and Backward Algorithm. This algorithm is used to train the Hidden Markov Model. Hidden Markov model is defined in a way that it represents the joint probability distribution over the set of hidden as well as the observed states that are random in nature. It makes an assumption that the k_{th} hidden variable when $(k-1)_{th}$ hidden variable is given is independent to all the previous hidden variables. This means that the current variable that is observed only depends on the current hidden state and is independent of the rest. The Baum–Welch algorithm incorporates the EM algorithm described earlier to find the maximum likelihood estimates of the hidden Markov model parameters when a certain data set is given.

Consider the following example which will help in understanding the Baum-Welch algorithm. In this example Baum Welch algorithm is implemented in terms of counts. We have a machine having two states, *q* and *r* whereas, *a* and *b* are the symbols.
Given the observation sequence there are a number of states that are possible by the machine.

String = abb aaa bbb aaa

Output Sequence $= q \xrightarrow{a} r \xrightarrow{b} q \xrightarrow{b} q \xrightarrow{a} r \xrightarrow{a} q \xrightarrow{a} r \xrightarrow{b} q \xrightarrow{b} q \xrightarrow{b} q \xrightarrow{a} r \xrightarrow{a} q \xrightarrow{a} r$

Source	Destination	Output	Counts
Q	R	A	5
Q	Q	B	3
R	Q	A	3
R	Q	B	2

Fig 5-4 Table of counts

Given the table of counts, we can calculate the probabilities of transition. For example the $q \xrightarrow{a} r$ transition has occurred 5 times in the output sequence and the total number of transitions from q being source state is 8 $(q \to r = 5 + q \to q = 3)$.

Hence,

$$P\left(q \xrightarrow{a} r\right) = \frac{5}{8} \tag{5.39}$$

Likewise,

$$P\left(q \xrightarrow{b} q\right) = \frac{3}{8} \tag{5.40}$$

This can be generalized as follows;

$$P\left(S^i \xrightarrow{w_k} S^j\right) = \frac{c\left(S^i \xrightarrow{w_k} S^j\right)}{\sum_{l=1}^{T}\sum_{m=1}^{A} c\left(S^i \xrightarrow{w_m} S^l\right)} \tag{5.41}$$

The above equation shows that transition from S^i to S^j with w_k is equal to count from S^i to S^j with output w_k divided by total number of counts with S^i as source. Where $c\left(S^i \xrightarrow{w_k} S^j\right)$ can be obtained by the following equation;

$$c\left(S^i \xrightarrow{w_k} S^j\right) = \sum_{s.n+1} P\left(S_{0,n+1} \middle| W_{0,n}\right) * n\left(S^i \xrightarrow{w_k} S^j, S_{0,n+1}, W_{0,n}\right) \tag{5.42}$$

The above equation shows that this method of taking the counts is valid if we have a single state sequence for an observation sequence. If we have multiple state sequences for an observation sequence, then we have to weigh the number of appearances by the probability of state sequence given the observation sequence $P\left(S_{0,n+1} \middle| W_{0,n}\right)$. For this we have to interplay between the two equations given above 4.18 and 4.19. Initially we will assume some transition probabilities and then the count is obtained from these probability values. We can also obtain the value of $P\left(S_{0,n+1} \middle| W_{0,n}\right)$ from transition probabilities which is nothing but Viterbi algorithm. Now from

the count we obtain new transition probabilities and from the new probability value we obtain the new count. Eventually after sometime the algorithm terminates when we see that there is no appreciable change in the probability values. This algorithm is called Expectation Maximization because we expect a value for the count and then maximize the probability of the observation sequence through this.

5.8.3.1 Baum-Welch Illustration

Baum Welch learns the probability values on arcs not the structure of Hidden Markov Model. This is a very important fact in machine learning. We will illustrate this with the help of example consisting of two states *q* and *r* and two symbols *a* and *b.*

Initially we have assumed the transition probabilities and then calculate count from these transition probabilities. Again from this count we will calculate new transition probabilities. This procedure is shown in the table shown below:

String: ababb

$\epsilon \to a$	$a \to b$	$b \to a$	$a \to b$	$b \to b$	$b \to \epsilon$	Path(P)	$q \overset{a}{\to} r$	$r \overset{b}{\to} q$	$q \overset{a}{\to} q$	$q \overset{b}{\to} q$
Q	R	Q	R	Q	Q	0.00077	0.00154	0.00154	0	0.00077
Q	R	Q	Q	Q	Q	0.00442	0.00442	0.00442	0.00442	0.00884
Q	R	Q	R	Q	Q	0.00442	0.00442	0.00442	0.00442	0.00884
Q	R	Q	Q	Q	Q	0.02548	0.0	0.0	0.05096	0.07644
Round Total						0.035	0.01	0.01	0.06	0.095
New Probab -ilities							0.06 (0.01/0.01+0.06 +0.095)	1.0	0.36	0.581

Table 5-5 One Run Baum Welch Algorithm Example

5.8.3.2 Computational Complexity of Baum Welch Algorithm

The computational part of the Baum Welch Algorithm is shown below. This gives us the mathematical illustration of this algorithm.

$$C\left(s^i \overset{w_k}{\rightarrow} s^j\right) = \frac{1}{P(W_{1,n})}[P(S_{1,n+1}, W_{1,n}) * n\left(s^i \overset{w_k}{\rightarrow} s^j, S_{1,n+1}, W_{1,n}\right) \quad (5.43)$$

$$P(S_{1,n+1}, W_{1,n}) * n\left(s^i \overset{w_k}{\rightarrow} s^j, S_{1,n+1}, W_{1,n}\right) =$$

$$\sum_{t=1}^{n} P(S_t = s^i, S_{t+1} = s^j, W_t = W_k, S_{1,n+1}, W_{1,n})$$
$$\sum_{t=1}^{n} \alpha_i(t) P(s_i \overset{w_k}{\rightarrow} s_i)\,, \beta_i(t+1) \quad (5.44)$$

CHAPTER 6

Wavelet Based Statistical Signal Processing

Statistical signal and image processing can be best carried out by using wavelet transform that finds its application in estimation, detection, classification and filtering. Existing wavelet based techniques do not take temporal correlations between wavelet coefficients into account. In this work, we have taken dependencies between wavelet coefficients into account and adapt to deal with non-Gaussian behavior by modelling them through new wavelet-based probability models.

6.1 2D - DWT

Wavelet transform of a video frame deals with its decomposition into a number of detail or wavelet coefficients $\{\psi^{LH}, \psi^{HL}, \psi^{HH}\}$ and one scaling coefficient ϕ^{LL}. This forms orthonormal basis for $L^2(R^2)$. An $N \times N$ image *z(t)* given a J-scale DWT can be decomposed as;

$$z(t) = \sum_{i \in Z^2} u_{J,i} \phi_{J,i}^{LL}(t) + \sum_{b \in B} \sum_{j=1}^{J} \sum_{i \in Z^2} w_{j,i}^{b} \psi_{j,i}^{b}(t) \qquad (6.1)$$

where $u_{J,i} = \int x(t) \phi_{j,i}(t) dt$ is the scaling coefficient and $w_{j,i}^{b} = \psi_{j,i}^{b}(t) dt$ represents the *(i)th* wavelet coefficient in j scale and sub-band $\mathcal{B}$.

$$\phi_{J,k,i}^{LL}(s,t) = 2^{-\frac{j}{2}} \phi(2^{-j}\ s-k, 2^{-j}\ t-i)$$

$$\psi_{J,k,i}^{LL}(s,t) = \ 2^{-\frac{j}{2}} \psi^{B}\ (2^{-j}\ s-k, 2^{-j}\ t-i)$$

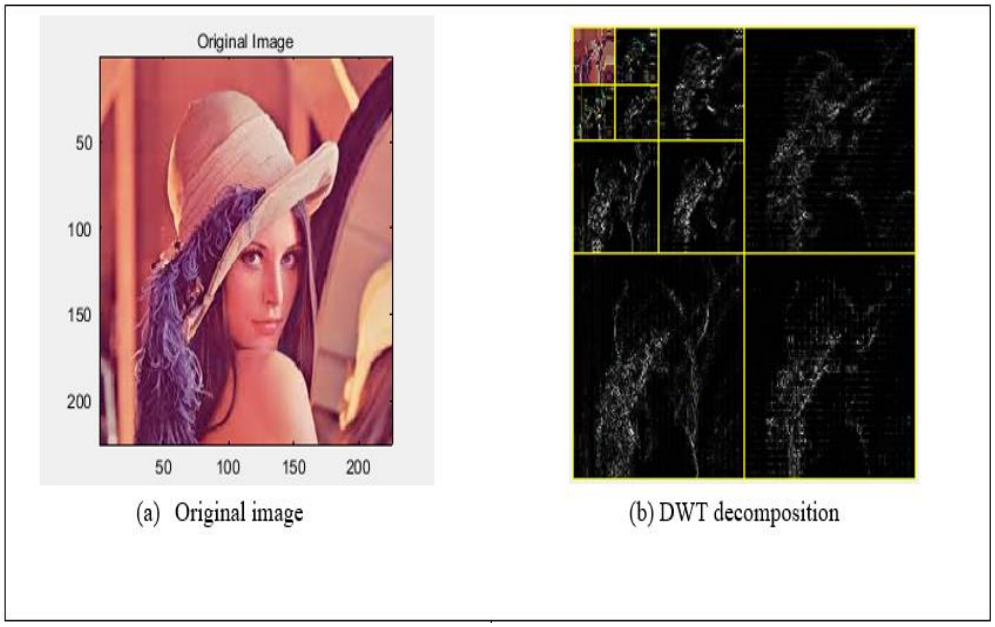


Fig. 6.1 Three level DWT decomposition

6.2 Modelling For Video Denoising Using Hidden Markov Model

Hidden Markov Model (HMM) captures the non-Gaussian statistics and complex dependencies among wavelet coefficients referred to as clustering property and persistence property respectively.

The Hidden Markov Model (HMM) uses a quad tree structure and has also been successfully used in Bandelet domain and contourlet domain. An m-state HMM links each wavelet coefficient with a hidden state variable in such a way that each wavelet coefficient is characterized by an m-dimensional state probabilities vector q and an m-dimensional standard deviation vector σ.

$$q = (q_1, q_2, ..., q_m)^t \qquad (6.2)$$

$$\sigma = (\sigma_1, \sigma_2, ..., \sigma_m)^t \qquad (6.3)$$

6.2.1 Capturing Non-Gaussian Densities

The non-Gaussian density of wavelet coefficients can be captured efficiently by Gaussian mixture model (GMM) and a multidimensional GMM is referred to as HMT. HMT models the wavelet coefficients as random variables having probability density function as a mixture of zero mean Gaussian distributions by means of a hidden state to designate small and large coefficient.

The pdf of wavelet coefficient *C* is defined as;

$$f_C(c) = \sum_{n=1}^{N} p_s(n)\, f_{C|S}(c|S=n) \qquad (5)$$

where $p_s(n)$ is probability mass function (pmf), S is the hidden state variable which is invisible and it controls the magnitude of wavelet coefficient.

The $f_{C|S}(c|S=n)$ is the conditional pmf given by the following eq;

$$f_{C|S}(c|S=n) = \frac{1}{\sigma_n \sqrt{2\pi}} \exp\left(- \frac{(b-\mu_n)^2}{2\sigma_n^2} \right) \qquad (6.4)$$

where μ_n and σ_n are the mean and variance respectively.

6.2.2 Capturing Dependencies

For capturing the interscale and intrascale dependencies among wavelet coefficients, HMT uses the probabilistic tree to model the Markovian dependencies between hidden states. To a wavelet decomposition of *J* scale and *K* sub-band, an HMT model contains the following parameters:

$P_{s_i}(n)$:pmf for the node root node S_i

$A_{j,k}$: a state transition probability matrix of *k* sub-band from scale *j-1* to scale *j*

$\mu_{j,k}, \sigma_{j,k}$: Gaussian mean and standard deviation vector of wavelet coefficient in *j* scale and *k* sub-band.

The state transition matrix shows $parent \rightarrow children$ state to state links between the hidden states that is given as;

$$A_{j,k} = \begin{bmatrix} p_{j,k}^{u \rightarrow u} & p_{j,k}^{u \rightarrow v} \\ p_{j,k}^{v \rightarrow u} & p_{j,k}^{v \rightarrow v} \end{bmatrix} \qquad (6.5)$$

$$\theta = [p(S_i = n), A_{j,k}, \mu_{j,k}, \sigma_{j,k}] \qquad (6.6)$$

where $p_{j,k}^{u\to u}$ or $p_{j,k}^{v\to v}$ represent the probability of a wavelet coefficient to be small or large given its parent is small or large. All these parameters are grouped together in the form of vector θ. It is to be noted here that each wavelet coefficient has different variances and state transition probabilities which lead to greater complexity in HMT model. We can reduce this computational complexity by a method of tying within scale [11]. According to this method, the wavelet coefficients have same density within a scale.

CHAPTER 7

Simulation and Results

Hidden Markov Tree based denoising technique in the perspective of 2D Gaussian Mixture Models (GMM) and 2D discrete wavelet transform (DWT) is used by applying it to each video frame independently. Expectation-Maximization (EM) algorithm iteratively finds the maximum likelihood of a fundamental distribution from a given data set. Our proposed method exploits effectiveness of DWT and the hierarchical relationships between its sub-bands.

7.1 Denoising Technique

7.1.1 Noisy Wavelet Coefficients

Let Q be a natural clean frame of a video sequence with $N \times N$ dimension and Q' be its noisy version such that $Q' = Q + E$ where E is zero mean white Gaussian noise. By performing wavelet decomposition on Q' the wavelet coefficient q' is obtained. Due to linearity of wavelet transform, we have;

$$q' = q + e \tag{7.1}$$

where q and e are the wavelet coefficients of Q and E respectively. We need to estimate the q given q'.

7.1.2 Model Parameters determination

HMT model is used to find a set of parameters $\theta_{q'}$. Initially, a two state GMM is used to characterize each wavelet coefficient and a noisy observation is used to initiate the HMT model. Then the interscale dependencies is captured by Markov tree and EM algorithm is used to obtain $\theta_{q'}$. According to [10], the added noise in a signal only increases its variance by leaving the other parameters unchanged. Hence, the noisy free observation θ_q can be extracted by fitting the HMT to the noisy observation and then subtracting the noise variance from it.

$$\left(\sigma_{(j,k,m),n}^{(q)}\right)^2 = \left(\left(\sigma_{(j,k,m),n}^{(q')}\right)^2 - \left(\sigma_{(j,k,m)}^{(e)}\right)^2\right)_+ \tag{7.2}$$

where j, k, m represent *j* scale, *k* sub-band and *n* state, *m-th* coefficient and $(g)_+ = g$ for $g \geq 0$ and $(g)_+ = 0$ for $g < 0$. Noise variance $\left(\sigma_{(j,k,m)}^{(e)}\right)^2$ can be estimated by median estimator in finest sub-band [22].

7.1.3 Clean Coefficients

Once θ_q is determined and state probability is given through HMT, we can get $E[q|q', \theta_q]$ by using Bayes estimator to get the clean coefficients;

$$\begin{aligned} q &= E[q|q', \theta_q] \\ &= \sum_n p(S|q, \theta_q) \times \frac{\left(\sigma_{(j,k,m),n}^{(q)}\right)^2}{\left(\sigma_{(j,k,m),n}^{(q')}\right)^2 + \left(\sigma_{(j,k,m)}^{(e)}\right)^2} q'_{j,k,m} \end{aligned} \tag{7.3}$$

Where j, k, m denote the *m-th* coefficient in scale j and sub-band k.

7.1.4 Reconstructed Frames

At the end, the inverse wavelet transform is applied to the obtained clean coefficients to get the reconstructed frames of a video sequence.

Figure shows step wise implementation of the proposed technique.

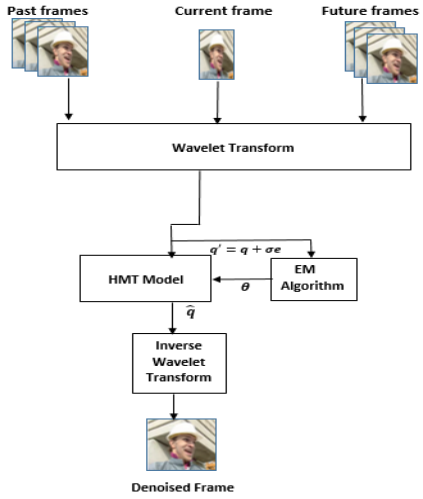
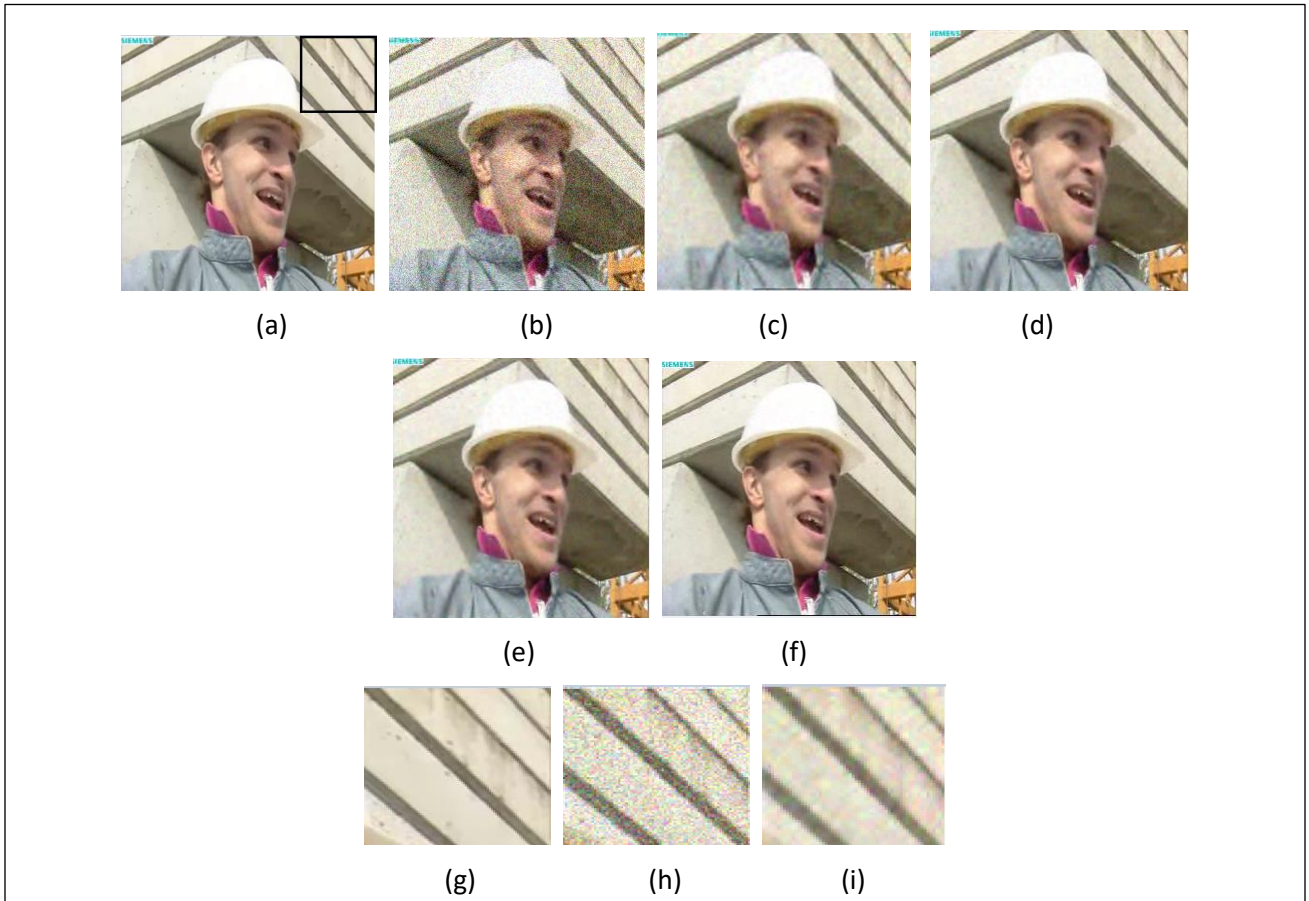


Fig.7.1 Diagram of proposed denoising process

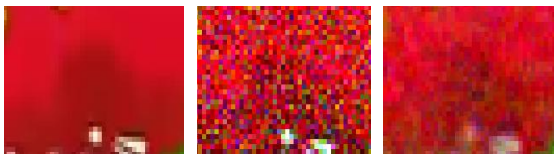
7.2 Simulation and Results

To illustrate the efficiency of proposed algorithm, different standard publicly available test video sequences are used such as BUS, MOBILE, SALESMAN, CHAIR, FOOTBALL and FOREMAN. Each sequence is artificially degraded with white Gaussian noise and speckle noise. The reconstructed frame is tested with the original one. Qualitative analysis is performed in Fig 7.2, 7.3 and 7.4 with existing state-of-the-art methods including NLM, VBM3D and CIFIC [18][17][24]. The block in original frame show zoomed regions and its comparison with other techniques.



(a) (b) (c) (d)

(e) (f)



(g) (h) (i)

(j) (k) (l)

Fig.7.4 Comparison of 58 frame of "BUS" with uniform Gaussian noise and its zoomed area shown by a block in original frame (a) Original image (b) Noisy image (c) NLM (d) V-BM3D (e) CIFIC (f) Proposed method (g) Zoomed area shown by a block in 'a' (h) Zoomed area of 'b' (i) Zoomed area of 'c' (j) Zoomed area of 'd' (k) Zoomed area of 'e' (l) Zoomed area of 'f'

The quantitative results are shown in Tables 7-1 to 7-4. Table 7-1 and 7-2 shows better performance of proposed algorithm in terms of Color PSNR (CPSNR), Mean Structural Similarity index (MSSIM) and Pearson's Correlation Coefficient (PCC).

Table 7-1. Quantitative Comparison of NLM, V-BM3D, CIFIC and proposed with uniform Gaussian noise. All CPSNR values are in decibel

Algorithm	Quantitative Measure	$\sigma_n = 15$					
		Video sequences					
		Foreman	Mobile	Bus	Chair	Football	Salesman
NLM	CPSNR	34.04	29.87	30.64	34.03	29.77	32.12
	PCC	0.896	0.854	0.886	0.954	0.951	0.950
	MSSIM index	0.678	0.713	0.810	0.801	0.543	0.884
V-BM3D	CPSNR	31.15	30.95	30.55	36.08	30.57	35.13
	PCC	0.940	0.899	0.951	0.968	0.959	0.951
	MSSIM index	0.794	0.798	0.824	0.804	0.577	0.896
CIFIC	CPSNR	32.13	31.96	33.47	36.40	31.59	35.18
	PCC	0.941	0.901	0.961	0.971	0.960	0.953
	MSSIM index	0.812	0.812	0.831	0.811	0.591	0.817
Proposed Algorithm	CPSNR	33.14	32.90	34.85	36.69	32.10	35.53
	PCC	0.959	0.971	0.977	0.974	0.962	0.967
	MSSIM index	0.877	0.839	0.862	0.882	0.673	0.830

Table 7-2. Quantitative Comparison of NLM, V-BM3D, CIFIC and proposed with uniform Gaussian noise. All CPSNR values are in decibel

Algorithm	Quantitative Measure	$\sigma_n = 20,25,30$					
		Video Sequences					
		Foreman	Mobile	Bus	Chair	Football	Salesman
NLM	CPSNR	33.09	28.17	27.82	31.07	27.32	29.03
	PCC	0.765	0.851	0.876	0.945	0.922	0.914
	MSSIM index	0.564	0.710	0.798	0.796	0.458	0.751
V-BM3D	CPSNR	30.11	29.15	27.81	33.91	27.80	32.13
	PCC	0.870	0.814	0.893	0.911	0.943	0.877
	MSSIM index	0.685	0.688	0.744	0.789	0.389	0.810
CIFIC	CPSNR	30.16	30.06	30.51	33.71	28.81	32.10
	PCC	0.876	0.866	0.899	0.854	0.912	0.945
	MSSIM index	0.723	0.765	0.743	0.712	0.412	0.789
Proposed Algorithm	CPSNR	32.14	31.60	31.13	33.89	29.35	33.23
	PCC	0.893	0.912	0.915	0.945	0.933	0.926
	MSSIM index	0.743	0.814	0.827	0.867	0.594	0.819

NLMC, CIFIC and V-BM3D do not consider speckle noise while proposed method outperforms the existing despeckling techniques. In Figure 7.5 and 7.6, the original frame corrupted with speckle noise is filtered by using different techniques including SOMA, MD, Wavelets [25][26][27] and proposed algorithm.

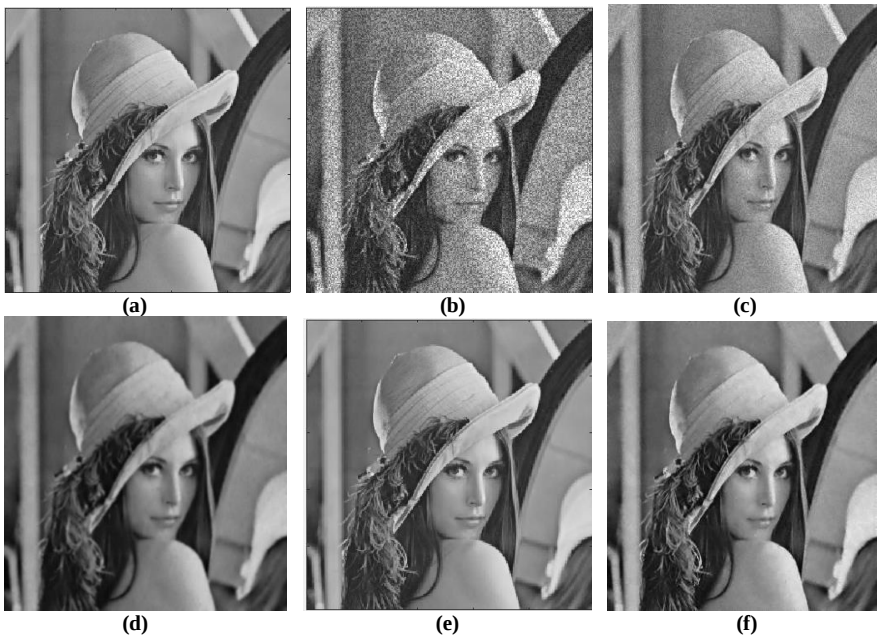


Fig.7.5. Qualitative Comparison of Proposed Algorithm with other techniques (a) Original Image (b) Noisy image (c) SOMA (d) MD (e) Wavelets (f) Proposed algorithm

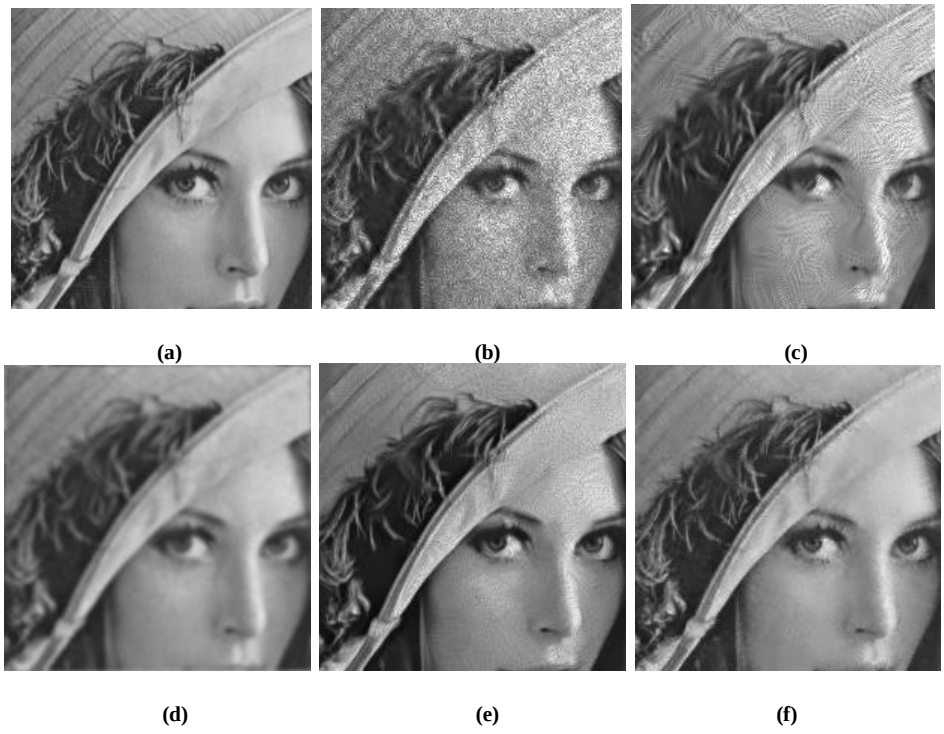


Fig.7.6. Qualitative Comparison of Zoomed Lena with other techniques (a) Original Image (b) Noisy image (c) SOMA (d) MD (e) Wavelets (f) Proposed algorithm

Table 7-3.Quantitative results of proposed algorithm with other techniques with Speckle noise and uniform noise level

Algorithm	Quantitative Measure	Test Images					
		Lena	Boat	Fruit	Building	Cameraman	Couple
SOMA	PSNR	26.22	28.09	26.18	28.16	27.25	28.07
	RMSE	137.02	132.028	130.714	133.93	135.47	131.35
	PCC	0.9521	0.9723	0.8725	0.9821	0.9617	0.8983
	MSSIM index	0.7163	0.7805	0.7678	0.7782	0.7852	0.9609
MD	PSNR	26.51	28.01	27.13	28.56	27.45	29.04
	RMSE	132.22	129.98	130.71	129.13	126.18	131.35
	PCC	0.9573	0.9745	0.9617	0.9894	0.9667	0.9941
	MSSIM index	0.7181	0.7962	0.7813	0.7890	0.7884	0.9621
Wavelet Denoising	PSNR	32.36	34.11	33.14	35.60	34.65	35.43
	RMSE	98.871	96.219	94.337	95.144	125.819	98.761
	PCC	0.9924	0.9933	0.9925	0.9889	0.9954	0.9981
	MSSIM index	0.9233	0.8785	0.8962	0.9045	0.8495	0.9233
Proposed Algorithm	PSNR	33.71	34.41	34.42	35.76	35.28	36.15
	RMSE	108.671	99.932	94.376	105.141	128.659	138.651
	PCC	0.9983	0.9964	0.9948	0.9991	0.9974	0.9987
	MSSIM index	0.9353	0.8789	0.9162	0.9154	0.8953	0.9451

Table 7-4. Quantitative results of proposed algorithm with different noise levels, the number of input frames = 3 and number of wavelet decomposition levels = 3

Quantitative Measures	Image Set	Speckle Noise				
		Frame 1=0.03 Frame 2=0.03 Frame 3=0.03	Frame 1=0.02 Frame 2=0.04 Frame 3=0.06	Frame 1=0.06 Frame 2=0.06 Frame 3=0.06	Frame 1=0.06 Frame 2=0.08 Frame 3=0.10	Frame 1=0.09 Frame 2=0.09 Frame 3=0.09
PSNR	Lena	33.714	32.412	31.644	31.174	30.135
	Boat	34.416	34.112	32.865	31.461	31.179
	Fruit	34.423	33.145	32.617	32.071	30.561
	Building	35.763	34.441	33.091	32.982	30.143
	Cameraman	35.286	33.982	32.981	32.132	30.013
RMSE	Lena	108.671	110.081	116.284	116.887	118.093
	Boat	99.932	103.012	104.016	105.018	105.091
	Fruit	94.376	95.714	95.976	96.019	97.023
	Building	105.141	106.109	108.512	108.854	109.158
	Cameraman	128.659	129.008	129.841	130.153	130.816
PCC	Lena	0.9983	0.9895	0.9851	0.9822	0.9813
	Boat	0.9964	0.9951	0.9948	0.9941	0.9916
	Fruit	0.9948	0.9940	0.9936	0.9927	0.9918
	Building	0.9891	0.9861	0.9823	0.9819	0.9811
	Cameraman	0.9974	0.9952	0.9923	0.9896	0.9881
MSSIM Index	Lena	0.9353	0.9351	0.9344	0.9342	0.9339
	Boat	0.8789	0.8775	0.8764	0.8732	0.8713
	Fruit	0.9162	0.9158	0.9151	0.9149	0.9143
	Building	0.9154	0.9147	0.9141	0.9138	0.9131
	Cameraman	0.8953	0.8951	0.8946	0.8941	0.8940

Table 7-3 and 7-4 presents comparison of proposed technique for speckle noise with existing state of-the-art algorithms. The comparison is shown in terms of peak signal-to-noise ratio (PSNR), Pearson's Correlation Coefficient (PCC), Root Mean Square Error (RMSE), and Mean Structural Similarity index (MSSIM). In addition, OCT image is also considered in the proposed algorithm and compared with wavelet as presented in Figures 5.7 and 5.8. The proposed method can suppress noise well as compared to other state-of-the-art algorithms and preserves edges efficiently.

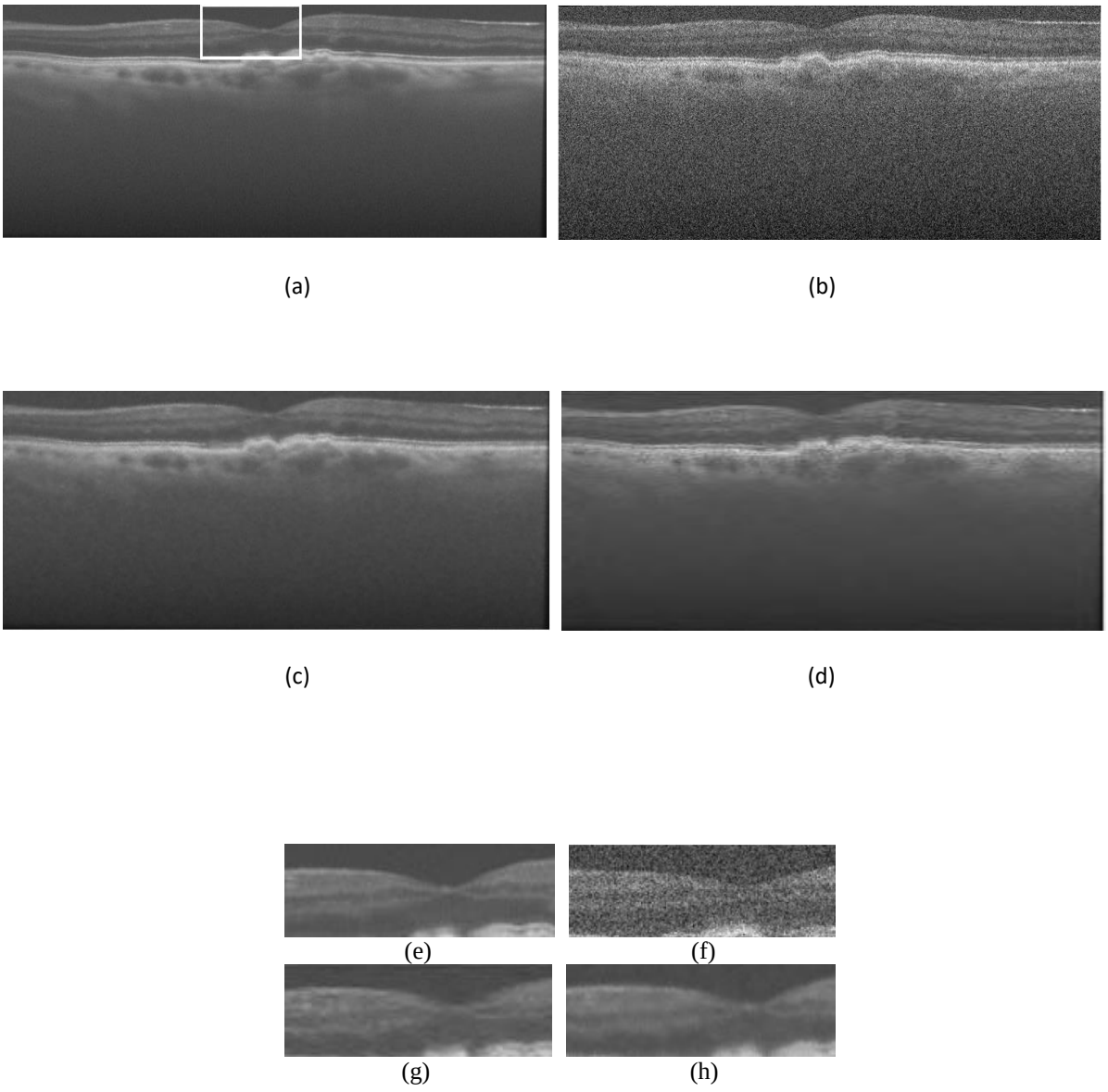


Fig. 7.7 Qualitative Comparison of proposed Denoising Algorithm on OCT Image (a) Average of human's eye frames (having least possible speckle noise) (b) a noisy frame (c) images denoised through wavelets (d) proposed algorithm using three noisy frames (e) Zoomed region displayed in 'a' (f) Zoomed region displayed in 'b' (g) Zoomed region displayed in 'c' (h) Zoomed region displayed in 'd'

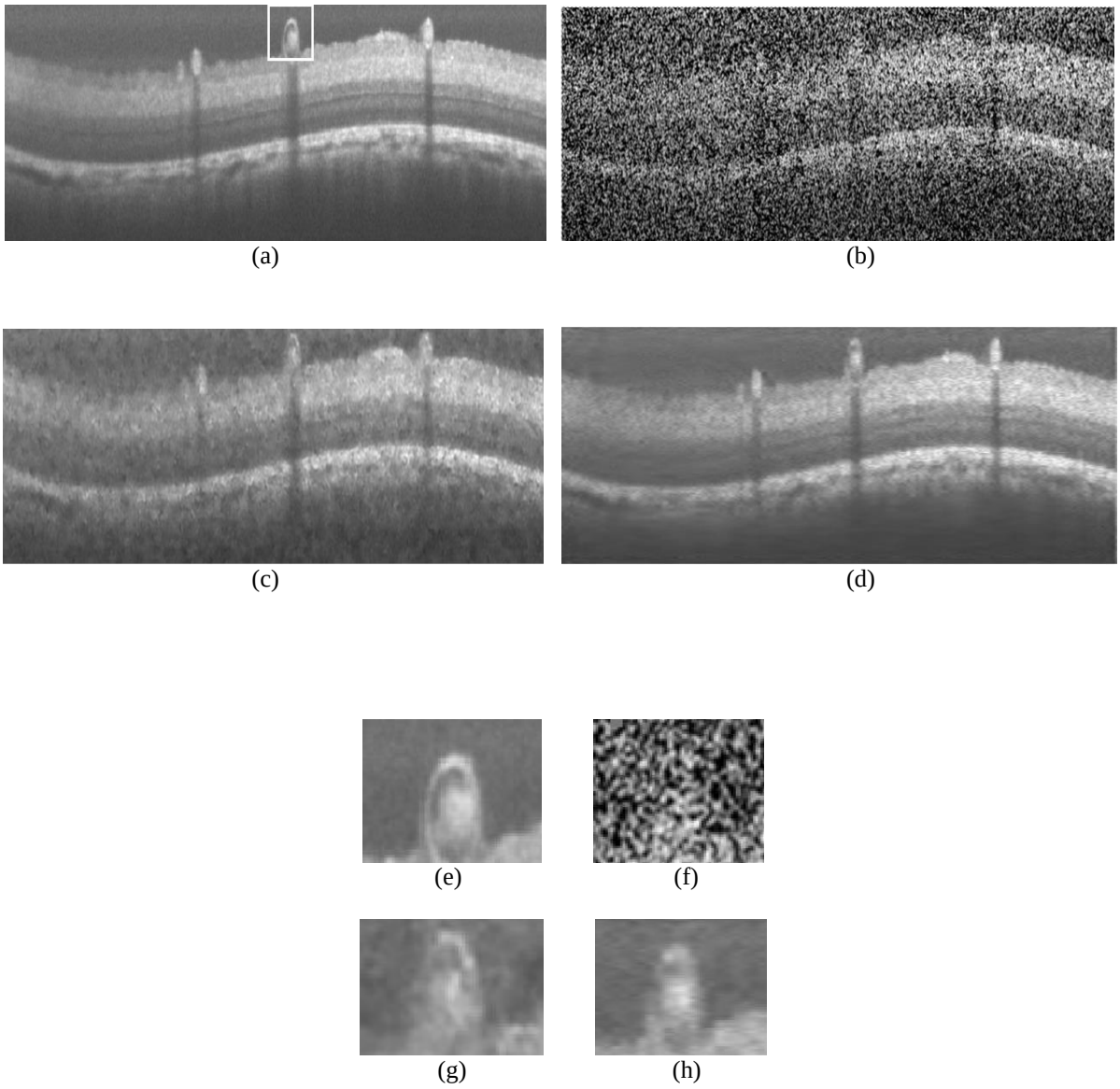


Fig. 7.8 Comparison on OCT Image (a) Average of pig's eye frames (having least possible speckle noise); (b) a noisy frame; (c) image denoised by wavelets (d) image denoised by proposed algorithm using three noisy frames (e) Zoomed block shown in 'a' (f) Zoomed block shown in 'b' (g) Zoomed block shown in 'c' (h) Zoomed block shown in 'd'

CHAPTER 8

Conclusion

Modeling is the essential part of any statistical image processing problem and for applications like estimation, detection, segmentation and denoising. Wavelet-based color video denoising is discussed in this thesis that is based on HMT and it captures the principle aspects of image configuration in wavelet domain. It is used in the framework of 2D-GMM and 2D-DWT scales and location. The primary properties of the wavelet transform locality, multi-resolution, and compression have led to powerful new approaches to statistical signal processing. However, conventional methods usually model the wavelet coefficients as statistically independent or jointly Gaussian. To cater the non-Gaussian nature of wavelet coefficients, Mixture densities have been incorporated and statistical dependencies between coefficients are captured by using probabilistic graphs/Tree.

We have trained HMT model using EM algorithm that provides a model that is flexible and has its fruitfulness in dealing with the wavelet coefficients to provide us with the modeling of different frames of a video sequence.

Experimental results have revealed that the proposed method outperforms the existing state-of-the-art techniques for color video sequences both in terms of qualitative and quantitative analysis. This method is capable of noise reduction and edge preservation.

8.1 Future Work

Although the proposed approach gives the encouraging denoising results, but there is always room for improvement to achieve better results. We can extend this technique to Synthetic Aperture Radar (SAR) images or high resolution images.

Moreover, we can add noise of different variances in different color components and perform joint denoising for all the color components. Motion estimation can be performed to denoise the static and moving regions separately.

This technique can be extended to other transform domains like Bandelet, Contourlet, Rigdelet and Curvelet.

Finally it is also possible to combine this technique with other methods to achieve better performance and to reduce the computational complexity and running time of the system.

Bibliography

1. Sharma, A., Singh, J., “Image denoising using Spatial domain filters: A quantitative study”, In proceedings of 6th International Congress on Image and Signal Processing (CISP) IEEE, Hangzhou (China), December 2013.

2. Narasimha, C., Rao, N,A., “Spatial domain filter for Medical Image enhancement”, In proceedings of International Conference on Signal Processing and Communication Engineering Systems IEEE, The Guntur (India), January 2015.

3. Wang, B., Xiong, Z., Zhang, D., et al. “Nonlocal image denoising via Collaborative Spatial-domain LMMSE estimation”, IEEE International Conference on Image Processing (ICIP), The Paris (France), October 2014.

4. Lee, J. S., “Digital Image Enhancement and Noise Filtering by Use of Local Statistics”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume. 2, No. 2, pp.165–168, 1980.

5. Li, X., Shen, H., Zhang, L., et al. “Sparse-based reconstruction of missing information in remote sensing images from spectral/temporal complementary information”, ISPRS Journal of Photogrammetry and Remote Sensing, Volume 106, pp. 1-15, 2015.

6. Ozkan, M. K., Sezan, M. I., Tekalp, A. M., “Adaptive motion-compensated filtering of noisy image sequences”, IEEE Transactions on Circuits and Systems for Video Technology ,Volume.3, No.4,pp. 277–290, 1993.

7. Liu, Y., Luo, Bing et al. “Weighting Wiener and Total Variation for Image Denoising”, Proceedings of the IEEE International Conference on Information and Automation Ningbo, China, August 2016.

8. Maggioni, M., Monge, E.S., Foi, A., “Joint removal of random and fixed-pattern noise through Spatiotemporal video filtering”, IEEE Transactions on Image processing, Volume. 23, No. 10, pp.4282-4296, 2014

9. Wang, X., Zhu, C., Li, S., et al. “Depth filter design by jointly utilizing spatial-temporal depth and texture information”, IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB ’15), The Ghent (Belgium), pp.1–5, June 2015.

10. Crouse, M. S., Nowak, R. D., Baraniuk, R. G., “Wavelet-based statistical signal processing using Hidden Markov Models”, IEEE Transactions on Signal Processing, Volume. 46, No.4 , pp.886-902, 1998.

11. Malfait, M., Roose, D., “Wavelet-based image denoising using a Markov Random Field a prior model”, IEEE Transactions on Image Processing, Volume.6, No.4, pp. 549-565, 1997.

12. Wen, B., Ravishankar, S., Bresler, Y., “Video denoising by online 3d-Sparsifying transform learning”, IEEE International Conference on Image Processing (ICIP), The Quebec City (Canada), pp. 1-5, September 2015.

13. Zhigang, D., Jingxuan, Z., Chunrong J., “An Improved Wavelet Threshold Denoising Algorithm”, Intelligent System Design and Engineering Applications (ISDEA), The Hong Kong (China), January 2013.

14. Riaz, M. U., Touqir, I., And Haider, M., “Wavelet-based image modelling for compression using Hidden Markov model”, International Journal of Advanced Computer Science and Applications, Volume.7, No.8, pp.1-7, 2016.

15. Ho, J., Hwang, W. L., “Wavelet Bayesian Network Image Denoising”, IEEE Transactions on Image Processing, Volume. 22, No. 4, 2014.

16. Wen, B., Ravishankar, S., Bresler, Y., “Video denoising by online 3d-Sparsifying transform learning”, IEEE International Conference on Image Processing (ICIP), The Quebec City (Canada), pp.1-5, September 2015.

17. Dabov, K., Foi, A., Egiazarian, K., “Video denoising by Sparse 3D transform-domain collaborative filtering”, In proceedings of 15th European Signal Processing Conference (EUSIPCO), The Poznan (Poland),2007.

18. Maggioni, M., Katkovnik, V., Egiazarian, K., et al. “Nonlocal transform-domain filter for volumetric data denoising and reconstruction”, IEEE Transactions on Image Processing, Volume. 22, No. 1, pp.119-133, January 2013.

19. Zuo, C., Liu, Y., Tan, X., et al. “Video Denoising Based on a Spatiotemporal Kalman-Bilateral Mixture Model”, The Scientific World Journal, pp. 1-10, 2013.

20. Aydin, V. A., Foroosh, H., “Motion-Compensated Temporal Filtering for Critically-Sampled Wavelet-Encoded Images” Cornell university library, Computer Vision and Pattern Recognition May 2017.

21. Hong-Zhi, W., Ling, C., Shu-Liang, S., “Improved video denoising algorithm based on spatial-temporal combination”, In Proceedings of the 7th International Conference on Image and Graphics (ICIG’13), IEEE, The Qingdao (China), pp.64-67, July (2013).

22. D. L, Donoho., I.M, Johnstone., “Ideal spatial adaptation via wavelet shrinkage”, Biometrika, Volume .81, pp. 425–455, 1994.

23. Haider, M., Touqir, I., Riaz, M. Usman., et al. “Denoising in Wavelet Domain Using Probabilistic Graphical Models”, International Journal of Advanced Computer Science and Applications (IJACSA), Volume .7, No. 11, 2016.

24. Dai, J., C. Au, O., Pang, C., And Zou, F., “Colour Video Denoising Based on Combined Interframe and Intercolor Prediction”, IEEE Transactions on Circuits and Systems for Video Technology, Volume. 23, No. 1, 2013.

25. Anupriya, A., Tayal, A., “Wavelet Based Image Denoising using Self Organizing Migration Algorithm”, CIIT International Journal of Digital Image Processing, Volume. 4, No.10, pp. 542-546, 2012.

26. Knaus, C., Zwicker,N., “Dual-Domain Image Denoising”, In Proceedings of IEEE International Conference on Image Processing, The Melbourne (Australia), pp. 440-444, September (2013).

27. Habib, W., Sarwar, T., Masood, A., et al. “Wavelet Denoising of Multi-frame Optical Coherence Tomography Data Using Similarity Measures”, IET Image Processing, Volume.11, No. 1, pp. 64-79, 2017.