CLASSIFICATION OF CPM SIGNALS USING EMPIRICAL MODE DECOMPOSITION AND APPROXIMATE ENTROPY

A Thesis in

Electrical Engineering

by

Jacob Kurien

© 2012 Jacob Kurien

Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Science

December 2012
The thesis of Jacob Kurien was reviewed and approved* by the following:

John F Doherty  
Professor of Electrical Engineering  
Thesis Advisor

George Kesidis  
Professor of Electrical Engineering

Kultegin Aydin  
Professor of Electrical Engineering  
Head of the Department of Electrical Engineering

*Signatures are on file in the Graduate School
ABSTRACT

In communication systems the most commonly used modulation scheme is the Continuous Phase Modulation. Finding the modulation index of the CPM signal is one of the most interesting problems in the field of Digital communication. In this thesis, a new method employing empirical mode decomposition and approximate entropy has been employed to determine the modulation index at the receiver end. The CPM signal received is first decomposed into individual intrinsic mode functions (IMFs) using empirical mode decomposition and the approximate entropy (ApEn) of these IMFs are calculated. ApEn gives a measure of the signal complexity and can be used as the feature in the classifier. Then we will employ classifiers to map the signal to its corresponding value of modulation index.

In this thesis we will also consider different classification methods like Linear Discriminant Analysis (LDA), Support vector machine (SVM), Probabilistic neural network (PNN) and find which classifier gives results with highest accuracy. Parameter selection for ApEn calculation has also been discussed at length while employing different classification techniques.
# TABLE OF CONTENTS

LIST OF FIGURES ........................................................................................................................................ vi

LIST OF TABLES............................................................................................................................................... viii

ACKNOWLEDGEMENTS ............................................................................................................................... ix

Chapter 1 Introduction to Classification of CPM signals................................................................. 1

1.1 Continuous Phase modulation ................................................................................................. 2
1.2 Representation of CPM signals .......................................................................................... 3
1.3 Pattern recognition applied to classification of modulation index of CPM signals ............. 5
1.4 Organization ......................................................................................................................... 6

Chapter 2 Empirical Mode Decomposition .................................................................................... 7

2.1 Introduction ............................................................................................................................. 7
2.2 Intrinsic mode functions ........................................................................................................ 8
2.3 Procedure ............................................................................................................................... 8
2.4 Implementation ...................................................................................................................... 11

Chapter 3 Approximate Entropy ...................................................................................................... 15

3.1 Logic behind Approximate Entropy ..................................................................................... 16
3.2 Algorithm to find Approximate Entropy ............................................................................. 16
3.3 Flowchart to find Approximate Entropy ............................................................................. 18
3.4 Variation of ApEn with r and m ......................................................................................... 19
3.5 Variation of ApEn with amplitude, frequency, phase and SNR ........................................ 21

Chapter 4 Classifiers ......................................................................................................................... 23

4.1 Linear Discriminant Analysis ............................................................................................... 23
4.2 Fisher’s Linear Discriminant for two classes ...................................................................... 27
4.3 Fisher’s Linear Discriminant for multiple classes .............................................................. 30
4.4 Probabilistic Neural Network ............................................................................................. 32
4.5 Support Vector Machine ..................................................................................................... 34

Chapter 5 Using EMD and ApEn to extract features for classification ........................................ 39

5.1 Time series generation .......................................................................................................... 39
5.2 Feature Extraction ................................................................................................................. 40
5.3 Monte Carlo Simulation ........................................................................................................ 41
5.4 Classification Process ........................................................................................................... 43
5.5 Sequential Forward Feature Selection .............................................................................. 44
Chapter 6 Results and Conclusions.................................................................46

  6.1 Linear Discriminant Analysis is used as the classification method.........46
  6.2 Probabilistic Neural Network is used as the classification method........52
  6.3 Support Vector Machine is used as the classification method..........58
  6.4 Comparison between the different classification systems (LDA, PNN and
      SVM)........................................................................................................64

Appendix Estimation of SNR .................................................................67
REFERENCES........................................................................................70
LIST OF FIGURES

Figure 1-1. The block diagram showing the role of automatic modulation classifier in a receiver. ................................................................. 2
Figure 1-2. Pulse shape for full response CPM signal. ................................................................. 4
Figure 1-3. Phase trajectory of a quaternary CPFSK [10]............................................................. 5
Figure 1-4. Block diagram showing pattern recognition approach [11] ..................................... 6
Figure 2-1. The EMD block diagram...................................................................................... 8
Figure 2-2. EMD algorithm [2] ........................................................................................... 10
Figure 2-3. The multi component data signal........................................................................... 12
Figure 2-4. IMF 1 ................................................................................................................. 12
Figure 2-5. IMF 2 ................................................................................................................. 13
Figure 2-6 IMF 3 ................................................................................................................. 13
Figure 2-7. Monotonic residue function res(t)......................................................................... 14
Figure 2-8. The reconstructed signal by adding all the IMFs with res(t).............................. 14
Figure 3-1. Algorithm of ApEn ........................................................................................... 18
Figure 3-2. ApEn variation with c parameter for m=1.......................................................... 19
Figure 3-3. ApEn variation with c parameter for m=2.......................................................... 20
Figure 3-4. ApEn variation with c parameter for m=3.......................................................... 20
Figure 3-5. (a) Variation of ApEn with Amplitude (b) Variation of ApEn with frequency (c) Variation of ApEn with phase (d) Variation of ApEn with SNR......................................... 22
Figure 4-1. Orientation of the decision surface with respect to the weight vector .............. 24
Figure 4-3. Decision region for multiclass Linear discriminant [21]........................................... 27
Figure 4-4. (a) The two classes are projected on the line shown and the corresponding overlapping histogram is also shown. (b) The projection in the second case is based on Fisher linear discriminant and the histogram suggests non overlapping projection on the new line. [21].................................................................................................. 28
Figure 4-5. Representation of probabilistic neural network [26] ......................................................... 34
Figure 4-6. Training the SVM to get the optimal hyper plane. [36] ....................................................... 35
Figure 4-7. The input space is mapped to a higher dimensional feature space [37] ............................... 37
Figure 5-1. CPM signal generation using Simulink............................................................................. 39
Figure 5-2. Block diagram representing the process of feature extraction ........................................ 41
Figure 5-3. Block diagram representing the process of Classification .................................................. 44
Figure 6-1. Accuracy Verses SNR for c=0.2 ....................................................................................... 47
Figure 6-2. Accuracy Verses SNR for c=0.3 ....................................................................................... 48
Figure 6-3. Accuracy Verses SNR for c=0.4 ....................................................................................... 49
Figure 6-4. Accuracy Verses SNR for c=0.5 ....................................................................................... 50
Figure 6-5. Accuracy Verses SNR for c=0.6 ....................................................................................... 51
Figure 6-6. Accuracy Verses SNR for different values of c parameter ............................................. 52
Figure 6-7. Accuracy Verses SNR for c=0.2 ....................................................................................... 53
Figure 6-8. Accuracy Verses SNR for c=0.3 ....................................................................................... 54
Figure 6-9. Accuracy Verses SNR for c=0.4 ....................................................................................... 55
Figure 6-10. Accuracy Verses SNR for c=0.5 ..................................................................................... 56
Figure 6-11. Accuracy Verses SNR for c=0.6 ..................................................................................... 57
Figure 6-12. Accuracy Verses SNR for different values of c .............................................................. 58
Figure 6-13. Accuracy Verses SNR for c=0.2 ..................................................................................... 59
Figure 6-14. Accuracy Verses SNR for c=0.3 ..................................................................................... 60
Figure 6-15. Accuracy Verses SNR for c=0.4 ..................................................................................... 61
Figure 6-16. Accuracy Verses SNR for c=0.5 ..................................................................................... 62
Figure 6-17. Accuracy Verses SNR for different values of c .............................................................. 63
Figure 6-18. Accuracy Verses SNR when different Classification techniques are used ............... 65
LIST OF TABLES

Table 6-1. For c parameter = 0.2 ................................................................. 47
Table 6-2. For c parameter = 0.3 ................................................................. 48
Table 6-3. For c parameter = 0.4 ................................................................. 49
Table 6-4. For c parameter = 0.5 ................................................................. 50
Table 6-5. For c parameter = 0.6 ................................................................. 51
Table 6-6. For c parameter = 0.2 ................................................................. 53
Table 6-7. For c parameter = 0.3 ................................................................. 54
Table 6-8. For c parameter = 0.4 ................................................................. 55
Table 6-9. For c parameter = 0.5 ................................................................. 56
Table 6-10. For c parameter = 0.6 ............................................................... 57
Table 6-11. For c parameter = 0.2 ............................................................... 59
Table 6-12. For c parameter = 0.3 ............................................................... 60
Table 6-13. For c parameter = 0.4 ............................................................... 61
Table 6-14. For c parameter = 0.5 ............................................................... 62
Table 6-15. For c parameter = 0.6 ............................................................... 63
I would like to thank my advisor Professor John F Doherty for guiding me throughout my stint in the graduate school. I am highly indebted to him for giving me an opportunity to learn and research under him. His ability to view the research problems from a higher perspective and solving them has instilled in me problem solving aptitude which will be very useful in my future endeavors. He was instrumental in the completion of this thesis in a timely manner. He played a key role in helping me make decisions of which courses to take to be successful in my career. These decisions eventually helped me get an internship and a full time job at Microsoft. He chalked a realistic plan for me and motivated me to stay focused.

I would like to express my heartfelt gratitude to Professor George Kesidis for being the member of the graduate committee.

A special thanks to the Electrical Engineering department at Pennsylvania State University for believing in my abilities and giving me an opportunity to be a part of this prestigious institute.

I would also thank my family for extending support when I needed it the most and motivating me to achieve success.
Chapter 1

Introduction to Classification of CPM signals

Automatic modulation classification finds great use in between the detection module and
the demodulation module of a receiver [6]. This is mainly used in military applications where we
need to jam all the signals of the enemy and receive the signals from the friendly force. It also
used in civilian applications. The software defined radio which deals with communication
between systems which might be completely different from each other can employ the automatic
modulation classifier techniques. Basically we are supplying additional information to the
demodulator which would be a great help in the classification of modulation. In other words, it
helps in creating intelligent communication system [8].

Automatic modulation classification is of two types:

1. Likelihood based approach
2. Feature based approach

In the likelihood based approach, the decision is taken by comparing the likelihood ratio against a
threshold. This approach basically minimizes the probability of wrong classification. The
disadvantage of this method is that computational complexity is very high. In the case of a feature
based approach, the features selected in an improvised manner are considered and the decision is
made on the observed value. The disadvantage of this approach is that it does not give the optimal
classification. The computational complexity is very less for this approach [6].
The block diagram showing where automatic modulation classifier fits in the receiver is shown in
figure 1-1. It consists of two units; one is the pre-processing module and the other is the classifier
algorithm. The pre-processing unit can be used to find the SNR value, etc. The use of the pre-
processing unit is not limited to that. The function of the pre-processing unit depends a lot on the
type of classification algorithm. The classifier algorithm provides the additional information to
demodulator which helps to reconstruct the transmitted signal without losing signal integrity.

Figure 1-1. The block diagram showing the role of automatic modulation classifier in a receiver.

1.1 Continuous Phase modulation

The past few years saw a boom in the field of digital communication. To operate each of
the communication system, we need channel bandwidth. Due to the increase in the number of
digital communication systems, Bandwidth became one of the most sought after commodity.
Hence there was a need for digital modulation schemes which made an efficient use of the
bandwidth during the data transmission. Continuous Phase Modulation (CPM) makes efficient
use of bandwidth and is one of the popular digital modulation techniques. CPM is a non-linear
digital modulation schema where the phase of the signal is forced to be continuous. Due to this
condition this modulation schema can be considered as modulation with memory. CPM is very
popular in wireless digital transmission because phase discontinuity in other modulation schemas
result in spectral spreading and spurious transmissions. It is highly robust to the non-linearities of
the amplifier [7]. Although CPM has high implementation complexity it is still considered as a
very good option during the designing of a communication system is because of power efficiency
and constant amplitude [9].
1.2 Representation of CPM signals

The continuous phase modulated signal which is transmitted can be represented by the following equation:

\[
s(t) = \sqrt{2E \over T} \cos(2\pi f_c t + \phi(t; I) + \phi_o)
\]  

(1)

In equation (1) \(s(t)\) is the CPM signal and \(\phi(t; I)\) represents the time varying phase of the carrier. \(\phi_o\) represents the initial phase of the carrier and \(f_c\) is the frequency of the carrier [10]. The carrier phase of CPM signals can be represented by equation (2).

\[
\phi(t; I) = 2\pi \sum_{k=-\infty}^{n} l_k h_k q(t - kT), \quad nT \leq t \leq (n + 1)T
\]

(2)

Here \(\{l_k\}\) is the sequence of M-ary information symbols selected from the alphabet set \(\pm 1, \pm 3, \ldots, \pm (M-1)\). \(\{h_k\}\) is the sequence of modulation indices, since we are considering that all the symbols in a sequence have the same modulation index hence the value of \(h_k\) can be replaced by \(h\). The modulation index \(h\) is twice the product of the peak frequency deviation \(f_d\) and the pulse duration \(T\).

\[h = 2f_d T\]

Let us consider that our pulse information is present in \(g(t)\) then \(q(t)\) can be represented as shown in equation (3).

\[q(t) = \int_0^t g(\tau)d\tau\]

(3)

The full response CPM satisfies the following condition \(g(t) = 0\) for \(t > T\) and the partial response CPM does not follow the above condition, that is \(g(t) \neq 0\) for \(t > T\) [10]. In our study we are only considering the full response CPM as our data signal and the rectangular pulse of duration 1 sec is used in modulation. LREC denotes a rectangular pulse of duration LT, where L is a positive integer. If the value of \(L = 1\) then this is the special case of CPM and it is known as CPFSK.
modulation. Since the \( g(t) \) is considered as a rectangular pulse the \( q(t) \) can be sketched as shown in Figure 1-2.

![Figure 1-2. Pulse shape for full response CPM signal.](image)

It will give a very good intuition of continuous phase modulation if we sketch the set of phase trajectories \( \phi(t; I) \) generated by all possible values of the Information sequence \( \{I_k\} \). Here we are interested in the CPM signal which is quaternary, that is \( M=4 \). In such a situation \( I_k \) can be \( \pm 1, \pm 3 \). The phase trajectory of a 4-ary CPM signal is shown in Figure 1-3. This is the result of Matlab simulation using CPM blockset in Simulink.
In this simulation, we have considered the modulation index as 0.5, pulse length as unity, M=4, and samples per symbol as 8. The number of symbols considered here is 5 and all possible combinations of the 5 symbols have been calculated to sketch the phase trajectory shown in Figure 1-3.

1.3 Pattern recognition applied to classification of modulation index of CPM signals

This is the approach followed during the study. We will derive the features of the preprocessed CPM signal and feed it to a classifier and try to find the modulation index of the CPM signal. The feature under consideration is Approximate Entropy values [11] and different
classification techniques like linear discriminant analysis, Support vector machine and Probabilistic neural networks are used and their accuracy has been compared with each other. The above mentioned classifiers are trained and they are fed with the features which are the ApEn values of the different IMFs in this study. All the above steps can be shown as a block diagram in Figure 1-4.

![Block diagram showing pattern recognition approach](image)

Figure 1-4. Block diagram showing pattern recognition approach [11]

1.4 Organization

The thesis is organized as follows:

- Chapter 2 introduces the concept of Empirical Mode decomposition
- Chapter 3 talks at length about the approximate entropy
- Chapter 4 explains the different types of classifiers used during the study
- Chapter 5 combines the concept of Empirical Mode decomposition and approximate entropy and extracts the features for an effective classification.
- Chapter 6 concludes the study
Chapter 2

Empirical Mode Decomposition

2.1 Introduction

Empirical Mode decomposition (EMD) is a non-linear technique developed by N. E. Huang in 1998 [1]. Most of the real world signals are non-linear and non-stationary; hence the EMD algorithm is a good way for the analysis of these signals. A signal might contain more than one oscillatory mode. The EMD algorithm will decompose the signal into individual components such that each component will have only one oscillatory mode. The empirically decomposed component is known as Intrinsic Mode Function (IMF). Contrary to all the methods for the analysis of the non-linear and non-stationary data, this new method is intuitive, direct, a posteriori and adaptive [3]. The reason this method is adaptive is because the basis of decomposition is based on, and derived from, the data [3]. Hence we are expanding the signal into a set of functions defined by the signal itself. Instantaneous frequency is the frequency of the signal at one time without taking the any information of the signal at any other time into consideration. The concept of instantaneous frequency should be considered while dealing with non-stationary signal. This takes into account the time varying nature of the process. For a multi-component signal the notion of a single-valued instantaneous frequency becomes meaningless and hence we will get a different value each time [4]. In such a case we can decompose a signal using EMD into IMFs with each IMF having its own instantaneous frequency.
2.2 Intrinsic mode functions

There are two conditions that need to be satisfied for any signal to be classified as an intrinsic mode function.

1. In the whole data set, the number of extrema and the number of zero crossings must either be equal or differ at most by one.

2. The mean value of the envelope defined by the local maxima and the envelope defined by the minima should be zero at any point.

The first condition is similar to the narrow band requirement for a stationary Gaussian process. The second condition is used to force the local symmetry. Hence the IMF, defined by the zero crossings, has only one mode of oscillation and hence the name intrinsic mode function [3].

2.3 Procedure

The EMD algorithm decomposes the input data signal into individual IMFs and residue. This can be represented using the following block diagram.

Figure 2-1. The EMD block diagram
Here the input signal is represented by $x(t)$ and the $k^{th}$ IMF is represented by $\text{imf}(k)$ and the residue is denoted by $\text{res}(t)$. The EMD can be shown mathematically as:

$$x(t) = \sum_{k=1}^{n} \text{imf}_k(t) + \text{res}(t)$$

Here we are assuming that the number of IMFs formed upon decomposition of the input signal $x(t)$ is “n”. The decomposition of the data signal into IMFs is based on three assumptions:

1. At least two extremas are present in the signal
2. The time span between the extremas will define the time scale
3. If the data does not have extremas but contains inflection points then the extremas are found by differentiating the data signal once or twice and the end result is obtained by the integration of the components [3].

The EMD algorithm consists of the following steps (Figure 2-1):

1. Find all the local maximas and local minimas for the given signal $x(t)$.
2. Then the maximas are interpolated to form the upper envelope function $\text{Max}(t)$ and the minimas are interpolated to form the lower envelope function $\text{Min}(t)$.
3. The local mean $m(t)$ of both the envelopes are found.

$$m(t) = \frac{\text{Max}(t) + \text{Min}(t)}{2}$$

4. This mean is subtracted from the actual signal $x(t)$. The resulting signal $h(t)$ will be a high oscillating pattern.

$$h(t) = x(t) - m(t)$$

5. Now check whether $h(t)$ is an IMF by checking for the two conditions mentioned in section 2.2.
6. If h(t) is not an IMF then continue all the steps from 1 to 5 with x(t) = h(t) until h(t) is an IMF. This iterative process is known as sifting [5].

Figure 2-2. EMD algorithm [2]
7. Once we get $h(t)$ which is an IMF calculate the residue $res(t)$.

8. Repeat the whole algorithm until we get a monotonic residue function $res(t)$ [2].

**2.4 Implementation**

Consider a data signal which has the highest frequency component as $\sin(6\pi t)$ and the lowest frequency component as $t$. Although the variation is linear we can consider this as the lowest frequency component. The data signal has two more sinusoidal components with frequency of $1/2$ and $1$. This multicomponent signal is as shown in Figure 2-3. This is decomposed using EMD algorithm to form 3 IMFs and a residue signal. Consider the following equation:

$$x(t) = \sum_{k=1}^{n} \text{imf}_k(t) + res(t)$$

Here the value of $n$ is 3 and the $res(t)$ is represented by the Figure 2-7 and the IMFs having single mode oscillation is represented by the Figure 2-4 to Figure 2-6. Consider Figure 2-4 which represents the IMF1 which contains the highest frequency and the frequency of the IMF decreases for IMF2 and IMF3.

The above equation can also be verified by adding up all the IMFs and the monotonic residue signal $res(t)$. So when we add all the signals shown in figure 2-4 to figure 2-7 we get the figure 2-8 which is same as the initial data signal $x(t)$ also represented in the figure 2-3.
Figure 2-3. The multi component data signal.

Figure 2-4. IMF 1
Figure 2-5. IMF 2

Figure 2-6 IMF 3
Figure 2-7. Monotonic residue function res(t)

Figure 2-8. The reconstructed signal by adding all the IMFs with res(t)
Chapter 3

Approximate Entropy

In our study we are not transforming the signal to use a different domain, we are keeping the time series signal in the time domain and hence we want to find a statistic to measure the chaos or the irregularity present in the signal which can directly operate on a time series data. In such a case the Approximate Entropy (ApEn) seems to be a very good measure of the signal complexity or irregularity [16]. The concept of Approximate Entropy was first introduced by Pincus [12] in 1991. The reason for popularity of this method in quantifying the complexity of a signal is due to the fact that it works well even if the data signal (sampled at equal intervals) is a relatively small time sequence. Now the question that might arise is why do we need this metric? Can’t we use statistics like mean and variance to distinguish two time series? To answer that question let us consider two time series of data.

Data 1 = \{1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2,1,2\}

Data 2 = \{1,2,2,2,1,2,1,2,1,2,1,2,1,2,1,2,2,2,1,1,1,2,2,2,2,2,2,1,1,1,2,2,2,1,1,2,1,1,2\}

The mean and variance of these two data sequence will be same. The sequence “Data 1” has values alternating between 1 and 2 and it is perfectly regular. The sequence “Data 2” has values 1 and 2 which occur randomly hence we need to quantify the irregularity in this signal and ApEn value seems to be the best fit in such cases [17].

ApEn delegates a positive real number to the time series. Even for a stochastic process, we will get a finite value [15]. A regular sequence of data will have a lesser value of ApEn whereas a more complex signal will have a higher ApEn value. The noisy signal can also be categorized as a complex signal and will have a relatively high ApEn value.
3.1 Logic behind Approximate Entropy

Here we will find the presence of repetitive pattern in time series. The concept followed here is if we can discern the presence of repetitive pattern then that time series is more predictable and hence the complexity being inversely proportional to predictability will also decrease. This will result in a lower ApEn value. ApEn reflects the probability of similar patterns of observation in the time series being followed by supplementary similar observations [17].

The Approximate Entropy statistics is represented as ApEn(m,r,N) where m is the embedding dimension and r denotes the threshold or the tolerance level. The ApEn will measure the logarithmic likelihood that the patterns of size m which are close considering the r as the tolerance level on successive comparisons.

3.2 Algorithm to find Approximate Entropy

To determine the approximate entropy of a time series we need to have previous knowledge of the two parameters:

- Embedding dimension (m)
- Level of filtering or the tolerance value (r) [14].

Steps are as follows:

1. Consider a time series data of length N. In this case the N data values are equally spaced in time. This can be represented by u(1), u(2), ………u(N).
2. Select the values of the two parameters m and r which are fed into the ApEn algorithm. The m represents the length of compared run of data and the filtering level is specified by r. According to [18]- [20] the recommended value of r is 0.2 – 0.3 times the standard
deviation of the time series signal. The value of m is also usually fixed at 1, 2 or 3 and the
input is usually a data sequence of length 100 to 5000 [13].

3. Create a sequence of vectors $X(1), X(2), \ldots, X(N-m+1)$ in $\mathbb{R}^m$ and this m-
dimensional space is defined by

$$X(i) = [u(i), \ldots, u(i+m-1)].$$

4. Now calculate $d[x(i), x(j)]$.

$$d[x(i), x(j)] = \max_{k=1,2,\ldots,m} [u(i + k - 1) - u(j + k - 1)]$$

5. Check whether $d[x(i), x(j)] \leq r$. Find the number of such $j's$ and save it in a variable
valLessThanThreshold.

6. The value $C_i^m(r)$ should be calculated for each $i$ such that $1 \leq i \leq N - m + 1$. This
value can be calculated by the following equation

$$C_i^m(r) = \frac{\text{valLessThanThreshold}}{N - m + 1}$$

7. After calculating $C_i^m(r)$ for all values of $i$ such that $1 \leq i \leq N - m + 1$, calculate

$$\phi^m(r) = \frac{\sum_{i=1}^{N-m+1} C_i^m(r)}{N - m + 1}$$

8. The Approximate Entropy (ApEn) is defined by

$$ApEn = \ln \phi^m(r) - \ln \phi^{m+1}(r).$$

Here $\ln(.)$ represents natural logarithm and the values of $m$ and $r$ are fixed in Step2 [13]
[17].
3.3 Flowchart to find Approximate Entropy

Figure 3-1. Algorithm of ApEn
3.4 Variation of ApEn with r and m

In this section we will increase the value of r from 0.1*standard deviation of the signal to 0.9 * standard deviation of the signal while keeping the value of m fixed. This experiment is conducted for m=1, 2 and 3. There are two data signals upon which this experiment is conducted; one is the sinusoidal signal of amplitude unity and frequency 1 Hz and the other signal is the former added with noise. The signal to noise ratio of the second signal is 10 dB. Here $r = c \ast \text{standard deviation of the signal}$. Hence we will vary the $c$ parameter.

Figure 3-2. ApEn variation with $c$ parameter for m=1
Figure 3-3. ApEn variation with c parameter for m=2

Figure 3-4. ApEn variation with c parameter for m=3
From the plots in Figure 3-2 to Figure 3-4 we can conclude that at higher value of the c parameter we will lose significant signal information and very small value of c will be very susceptible to noise. These figures also tell that as the value of m is increased the value of c parameter which provides the maximum ApEn value also increases. In other words the maximum ApEn value for different values of the c parameter gets shifted towards the right and the relative magnitude of the ApEn in comparison with the lower values of m gets reduced.

### 3.5 Variation of ApEn with amplitude, frequency, phase and SNR

The variation of ApEn with amplitude is shown in Figure 3-5 (a). Here we are considering a sinusoidal signal and keeping its frequency and phase constant and we will change the amplitude linearly from 1 to 20. Figure 3-5 (a) shows that the ApEn value will not be affected with the change in amplitude. Hence we can say that the complexity of a signal does not vary when its amplitude is varied. Figure 3-5 (b) represents the change of ApEn value with the change in frequency. Here also we are considering a sinusoidal signal and fixing the amplitude and phase and increase the frequency of the signal from 1Hz to 15 Hz. It has been observed that the ApEn value increases as the frequency of the signal is increased. In other words, at higher frequency the complexity or unpredictability of the signal is more. The variation of ApEn with the phase of the sinusoidal signal is shown in Figure 3-5 (c). The variation in phase does not affect the complexity or the ApEn value of the signal. Now to find the variation of the ApEn value with the signal to noise ratio, we add additive white Gaussian noise in our signal. The amplitude, frequency and phase of the signal were kept as a constant and the SNR value was varied from 1 to 20 dB to get the Figure 3-5 (d). According to the Figure 3-5 (d) as the noise level in the signal increases i.e. the SNR value decreases, the ApEn value increases. For all these experiments the parameters of the
ApEn algorithm was also kept fixed. The tolerance level was set at 0.2 times the standard deviation of the signal and the embedding dimension “m” was set at 2 [11].

Figure 3-5. (a) Variation of ApEn with Amplitude (b) Variation of ApEn with frequency (c) Variation of ApEn with phase (d) Variation of ApEn with SNR
Chapter 4

Classifiers

In a usual classification problem we have K classes and we would like to map an input \( x \) to one of the K classes. The class is defined by \( C_k \) where \( k = 1, 2, \ldots, K \). The whole space is divided into K classes with the help of decision boundary or decision surface and the input \( x \) is classified into one of the K classes [21].

4.1 Linear Discriminant Analysis

Now let’s focus our attention on the linear model for classification. In the case of a linear model the decision surface are categorized by the linear function of the input vector. Here we are assuming the input \( x \) is a D dimensional vector. Hence we will need \((D-1)\) dimensional hyper planes to divide the space into classes using the decision surfaces. If the data is separable among classes then we call it linearly separable.

\[
y(x) = f(w^T x + w_0)
\]

(1)

The function \( f(.) \) used in equation (1) is also known as the activation function. It should be noted that the activation function is a nonlinear function. Though the activation function is a nonlinear function still the decision surface is a linear function of the input vector. The decision surface can be represented by \( y(x) = constant \).

Hence we have, \( w^T x + w_0 = constant \)

(2)

This proves that the decision surface is a linear function of \( x \). The discriminant function is defined as a function that takes the input vector \( x \) as its input and assigns it to a class \( C_k \).
Let’s first consider the case with 2 classes as it will be easier to derive and then we will generalize all equation for the case of multiple classes. For a two class system the simple linear discriminant function can be represented as

$$y(x) = w^T x + w_0$$  \hspace{1cm} (3)

In the above equation $w$ is the weight vector and $w_0$ is the bias. The negative of this value is also known as threshold in this context. The condition governing the separation of the space into different classes is as follows:

If $y(x) \geq 0$ then $x$ belongs to the class $C_1$, else $x$ belongs to the class $C_2$.

Consider two points $x_1$ and $x_2$ on the decision surface. Hence $y(x_1) = y(x_2) = 0$.

$$w^T (x_1 - x_2) = 0$$  \hspace{1cm} (4)

From equation (4) we can infer that the weight vector is orthogonal to all the vectors lying on the decision surface. In other words the orientation of the decision surface is decided by the weight vector [21].

---

Figure 4-1. Orientation of the decision surface with respect to the weight vector
The distance of the decision boundary from the origin is decided by \( w_0 \). Let the input be represented by \( x \) and the perpendicular distance of \( x \) from the decision boundary is given by \( r \).

Let us assume that the vector from origin to the projection of the input vector on the decision plane be given by \( x_\perp \). This is also represented in Figure 4-2.

From Figure 4-2 we have,
\[
x = x_\perp + \frac{r \cdot w}{\|w\|} \tag{5}
\]
\[
y(x) = y(x_\perp) + y(\frac{r \cdot w}{\|w\|}) \tag{6}
\]

Substituting equation (3) in equation (6), we get
\[
w^T x = w^T \frac{r \cdot w}{\|w\|}
\]

Here we have \( y(x_\perp) = 0 \) since \( x_\perp \) is present on the decision surface.

Finally, \( r = \frac{y(x)}{\|w\|} \) \tag{7}

Till now we have discussed the case where the number of class is two. Now consider the case where the number of classes is greater than 2 i.e. \( K>2 \). We will have \( K-1 \) classifiers each.
solving a two class problem in which it compares itself against the rest. This is also known as the
one verses the rest classifier. But the disadvantage is that we might encounter some region in the
decision space which is not uniquely the part of a single class. Another way to solve this problem
is to use \( \binom{K}{2} \) two class classifiers. This is known as one versus one classifier. But this method
also suffers from the issue of ambiguous region. To solve the above mentioned issue we will use
a K class discriminant comprising of K linear functions [21]. This can be represented in the
equation (8).

\[
y_k(x) = w_k^T x + w_{k0}
\]  

(8)

So if we are given an input \( x \), we can say that the input belongs to a class \( C_k \) if and only if the
following condition is met for all \( j \neq k \).

\[
y_k(x) > y_j(x)
\]  

(9)

We can also infer from the above condition that at the decision boundary of \( C_k \) and \( C_j \) we have
\[ y_k(x) = y_j(x) \]

The (D-1) dimensional hyper plane is defined by

\[
(w_k - w_j)^T x + w_{k0} - w_{j0} = 0
\]

Now let’s focus on the decision region in case of a multiclass linear discriminant. This situation is
shown in the Figure 4-3. We are considering two points \( x_1 \) and \( x_2 \) to be present in the class \( C_k \).
The distance between these two points \( x_1 \) and \( x_2 \) is given by \( \bar{x} \). Hence \( \bar{x} \) can be represented as

\[
\bar{x} = \lambda x_1 + (1 - \lambda) x_2
\]

(10)

where \( 0 \leq \lambda \leq 1 \). Since the decision surfaces are the linear function of \( x \) so using equation (10) we
can write the following relation

\[
y_k(\bar{x}) = \lambda y_k(x_1) + (1 - \lambda) y_k(x_2)
\]

Since \( x_1 \) and \( x_2 \) lie in the class \( C_k \), from equation (9) we have;
\[ y_k(x_1) > y_j(x_1) \text{ for all } j \neq k \] and
\[ y_k(x_2) > y_j(x_2) \text{ for all } j \neq k. \]

Hence we can conclude that \( y_k(\bar{x}) > y_j(\bar{x}) \) for all \( j \neq k \) \hspace{1cm} (11)

From equation (10) we can conclude that \( C_k \) is singly connected and also convex [21].

4.2 Fisher’s Linear Discriminant for two classes

Let’s consider the case of two classes and the input is of dimension D. Fisher’s Linear Discriminant is based on the principle of dimensionality reduction. In this case the D dimensional input is reduced to a single dimension since \( K=2 \).

\[ y(x) = w^T x \]

Now the projection of a D dimensional data on a single dimension might lead to the loss of information. Also, the well separated classes in D dimension could be overlapping when they are
projected to a single dimension. Hence we can maximize the separation between the classes by adjusting the weight vector \( w \). Let the two classes be denoted by \( C_1 \) and \( C_2 \) and the number of points in these classes be \( N_1 \) and \( N_2 \) respectively. The mean of the two classes is represented by \( m_1 \) and \( m_2 \).

\[
\begin{align*}
m_1 &= \frac{1}{N_1} \sum_{n \in C_1} x_n \\
m_2 &= \frac{1}{N_2} \sum_{n \in C_2} x_n
\end{align*}
\] (12)

To increase the separation between the two classes we have to maximize when \( m_2 - m_1 \) is projected.

Hence maximize \( m_2 - m_1 = w^T(m_2 - m_1) \) (13)

where \( m_k = w^Tm_k \) which is the mean of projected data from the class \( C_k \).

The magnitude of the weight vector \( w \) is fixed to have a unit length and we will do constrained maximization on the whole expression.

In other words, \( \sum_n w_n^2 = 1 \) is satisfied in the above case [21].

![Figure 4-4](image.png)

Figure 4-4. (a) The two classes are projected on the line shown and the corresponding overlapping histogram is also shown. (b) The projection in the second case is based on Fisher linear discriminant and the histogram suggests non overlapping projection on the new line. [21]
In Figure 4-4 (a) we have two classes which are well separated in the original dimension but when they are projected on the line which is parallel to the line joining their individual class means then they overlap with each other as shown in the histogram. Blue color is used for representing one class and the other class is represented by red color and the overlap when the dimension is reduced is shown by the histogram bars having both red and blue color. In the Figure 4-4 (b) the projection is based on the Fisher’s Linear Discriminant which aims at increasing the class separation. In order to increase the class separation the inter class variance is maximized and the intra class variance is minimized. In other words we will be maximizing the between class variance and minimizing the within class variance.

The within class variance of a class $C_k$ is given by

$$s_k^2 = \sum_{n \in C_k} (y_n - m_k)^2$$

where $y_n = w^T x_n$.

Total intra class variance can be given as $s_1^2 + s_2^2$ since we have only two classes $C_1$ and $C_2$ and we have to minimize this quantity. The inter class variance for 2 classes $C_1$ and $C_2$ whose individual class means are $m_1$ and $m_2$ is given by $(m_2 - m_1)^2$ and we have to maximize this term. Fisher criterion, denoted by $f(w)$, is the ratio of the inter class variance and intra class variance. The Fisher criterion needs to be maximized to get non-overlapping projection. Fisher criterion $f(w)$ can be represented mathematically as shown below:

$$f(w) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$$

Let $S_B$ be the inter class covariance matrix and $S_w$ be the intra class covariance matrix. They can be represented as follows:

$$S_B = (m_2 - m_1)(m_2 - m_1)^T$$

$$S_w = \sum_{n \in C_1} (x_n - m_1)(x_n - m_1)^T + \sum_{n \in C_2} (x_n - m_2)(x_n - m_2)^T$$
Hence the Fisher criterion in equation (14) can be rewritten using equation (15) and (16) as shown below:

$$J(w) = \frac{w^T S_{gw} w}{w^T S_{gw} w}$$

(17)

To maximize $J(w)$, differentiate $J(w)$ with respect to $w$ and equate it to 0. The condition which maximizes $J(w)$ is represented in equation (18).

$$(w^T S_{gw} w) S_{gw} w = (w^T S_{gw} w) S_{gw} w$$

(18)

In the equation (18), the terms $w^T S_{gw} w$ and $w^T S_{gw} w$ are scalars also $(m_2 - m_1)$ is a vector in the same direction as $S_{gw} w$. Multiply both sides with $S_{gw}^{-1}$ to get

$$w = S_{gw}^{-1} (m_2 - m_1)$$

(19)

The Fisher linear discriminant can be represented by equation (19), so we will chose a threshold $y_k$ and for every new data point we will check whether $y_k \geq y_0$. If this condition is met then we can conclude that the data point belongs to the class $C_1$ or else it belongs to $C_2$ [21].

**4.3 Fisher’s Linear Discriminant for multiple classes**

Let’s generalize the results found in the previous section for the case of multiple classes i.e. $K > 2$. Here the D dimensional input data points are taken into consideration and it is assumed that $D > K$. Here we are presenting $D’$ new features which are linear and can be represented as

$$y_k = w_k^T x \quad \text{where } k = 1, \ldots, D'$$

They can be collectively written as

$$y = W^T x$$

(20)

In equation (20) the weight matrix $W$ is formed by using the weight vectors as the columns of the matrix. Also note that we are not using any bias while defining $y$. The matrix dimension of $W^T$ will be $K-1 \times D$ and that of $x$ is $D \times 1$. Hence $y$ has a dimension of $K-1 \times 1$. 


The next task is to find the within covariance matrix also known as the intra class covariance matrix. If \( S_k = \sum_{n \subset c_k} (x_n - m_k)(x_n - m_k)^T \) where \( m_k = \frac{1}{N_k} \sum_{n \subset c_k} x_n \) then the within covariance matrix can be given as

\[
S_w = \sum_{k=1}^{K} S_k
\]  \((21)\)

In the above equation, \( N_k \) is the total number of instances in the class \( c_k \). The total covariance matrix is represented by \( S_T \) and denoted mathematically as shown in equation (22).

\[
S_T = \sum_{n=1}^{N} (x_n - m)(x_n - m)^T
\]  \((22)\)

Here \( N \) is the total number of instances in the space and \( m \) is the mean of the total data set.

\[
m = \frac{1}{N} \sum_{n=1}^{N} x_n = \frac{1}{N} \sum_{k=1}^{K} N_k m_k
\]

And hence the total covariance matrix is given by

\[
S_T = S_w + S_B
\]  \((23)\)

where \( S_B = \sum_{k=1}^{K} N_k (m_k - m)(m_k - m)^T \)

The above covariance matrix was in the original \( D \) dimensional space, now let’s take the \( D' \) dimensional projected space “\( y \)” into consideration. So the new intra covariance and inter covariance matrix is as shown below:

\[
s_w = \sum_{k=1}^{K} \sum_{n \subset c_k} (y_n - \mu_k)(y_n - \mu_k)^T
\]  \((24)\)

\[
s_B = \sum_{k=1}^{K} N_k (\mu_k - \mu)(\mu_k - \mu)^T
\]  \((25)\)

where \( \mu_k \) and \( \mu \) are given by the following equation

\[
\mu_k = \frac{1}{N_k} \sum_{n \subset c_k} y_n \quad \text{and} \quad \mu = \frac{1}{N} \sum_{k=1}^{K} N_k \mu_k
\]

And the Fisher criterion is given by equation (26) where \( S_w \) and \( S_B \) is the one given in equations (24) and (25).

\[
f(w) = \frac{S_B}{S_w}
\]  \((26)\)
4.4 Probabilistic Neural Network

The probabilistic neural network is used for a multi class classification where we already have some of the points belonging to each class and we are provided with a data sample which is to be classified to fall in one of the classes. This neural network implements a statistical algorithm known as kernel discriminant analysis.

The theory of classification in case of a probabilistic neural network is that if we know the probability density function of each of the data sets representing the whole class, then it can be found whether the data sample \( x \) belongs to a particular class \( \text{“i”} \) if the following condition is satisfied:

\[
f_i(x) > f_k(x) \text{ for all } k \neq i.
\]

Here \( f_i \) represents the probability density function of class \( \text{“i”} \) and \( f_k \) represents the probability density function of class \( \text{“k”} \). The other parameters that come to play are priori probability and cost of misclassification [26]. If \( h_k \) represents the priori probability that the data sample \( x \) belongs to class \( k \) and \( h_i \) represents the priori probability that the data sample \( x \) belongs to class \( i \). Also let’s assume that the cost of misclassification of \( x \) in class \( i \) be given by \( c_i \) and the cost of misclassification of \( x \) in class \( k \) is given by \( c_k \). Then according to the Bayes optimal decision rule the data sample \( x \) belongs to class \( i \) if the following condition is satisfied for all \( i \neq k \).

\[
h_i c_i f_i(x) > h_k c_k f_k(x)
\]

At the decision boundary we have \( h_i c_i f_i(x) = h_k c_k f_k(x) \) [26] [27]. As the training set increases, the probabilistic neural network converges to a Bayes optimal classifier since the estimated density function approaches to the actual density function assuming the latter is smooth. Each data point corresponds to a pattern unit which can be represented by the Gaussian density function with the peak centered on the data point’s location. Consider a vector input \( X \) to
the network. The input is assumed to be multivariate in this case [27]. Hence the density function for one sample in the class can be given by

\[
f'(X) = \frac{1}{(2\pi)^{p/2} \sigma^p} e^{-\frac{||x-x_k||^2}{2\sigma^2}}
\]  

(27)

In equation (27) the \(X\) represents the unknown input, \(X_k\) represents the position of the \(k^{th}\) sample in the class, \(\sigma\) is the soothing parameter and \(p\) gives the length of the vector.

Using equation (27) we can find the pdf of the whole class \(C_i\) as shown in equation (28).

\[
f_i(X) = \frac{1}{(2\pi)^{p/2} \sigma^p n_i} \sum_{k=1}^{n_i} e^{-\frac{||x-x_k||^2}{2\sigma^2}}
\]  

(28)

So the data sample \(X\) belongs to class \(C_i\) if the following condition is satisfied

\[f_i(X) > f_k(X)\] for all \(i \neq k\).

The probabilistic neural network classification procedure maps into a feed-forward neural network structure consisting of four layers, namely:

1. Input layer
2. Pattern layer
3. Summation layer
4. Output layer

The network structure of a PNN is represented in Figure 4-5. The input layer represents the data sample \(X\) and the pattern layer finds the \(f'(X)\) shown in equation (27). The summation layer sums all the \(f'(X)\) for a particular class. Finally the output layer finds the maximum of all the values found in the summation layer and makes the decision as to which class the data sample \(X\) belongs.

Some of the advantages of using a PNN is that the training of the classifier is done at a fast rate in comparison to other classification techniques like back propagation. It has a parallel structure and it converges to a Bayes optimal classifier for large number of data set. One of the disadvantages of using PNN is that it requires a large memory for its implementation.
4.5 Support Vector Machine

Support Vector machine is one of the most popular tools for classification in the field of machine learning. They have been used in a variety of real-world applications like face recognition [29], digital protection of power transformer [30], Robotics and Control System [31], Handwriting recognition [32], Biomedical applications [33] [34], etc. Let’s first consider the case of a two-class linear classifier. The training vectors are denoted by \( x_i \) where \( i = 1, 2, \ldots, l \) and the known classes can be found by defining an indicator vector \( y \).

\[
y_i = \begin{cases} 
1 & \text{if } x_i \text{ in class 1} \\
-1 & \text{if } x_i \text{ in class 2}
\end{cases}
\]  \hspace{1cm} (29)
Here each $x_i$ can be multi-dimensional depending on the dimensions of the feature space [35].

The aim while we train a Support Vector machine is to maximize the classifier margin. The training points which are shown with solid circles and squares, in figure 4-6, are the support vectors which determine the classifier margin. This is the projection view of the hyper plane in the two dimensional space. Support Vectors are responsible for stopping the margin and the classifier margin can be defined as the width to which a decision boundary can be extended before hitting a data point. Linear SVM also commonly denoted as LSVM is the maximum margin linear classifier. We can infer from Figure 4-6 that only the support vectors are important and the rest of the training data can be ignored. In case of a SVM our goal is to maximize the classifier margin.

The separating hyper plane is given by equation (30).

$$w^T x_i + b = 0$$  \hspace{1cm} (30)

The decision function is defined by the following equation

$$f(x) = \text{sgn}(w^T x + b)$$  \hspace{1cm} (31)
Distance between $\mathbf{w}^T \mathbf{x}_i + b = 1$ and $\mathbf{w}^T \mathbf{x}_i + b = -1$ which when maximized gives the maximum classifier margin.

$$d = \frac{2}{||\mathbf{w}||}$$  \hspace{0.5em} (32)

Maximizing $d$ in equation (32) is equivalent to minimizing $d'$ in equation (33)

$$d' = \min \left( \frac{1}{2} \mathbf{w}^T \mathbf{w} \right)$$  \hspace{0.5em} (33)

subjected to the constraint $y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1$ for $I = 1, 2, \ldots, I$ [38]

We should allow for some training errors and this will help us in building a generalized classifier which will perform better with the testing data. In other words, we have to find the tradeoff between the good classification of the training data set and the good classification of future testing data set. To get a 100 percent training accuracy might not be beneficial for classifying the future data sets and this is known as overfitting of the training data. To prevent overfitting we must add a cost factor $C \sum_{i=1}^I e_i$ in our minimization function which will allow for the training errors. The term $\sum_{i=1}^I e_i$ will prevent from having too many errors. Hence the minimization function can be rewritten as

$$\min \left( \frac{1}{2} \mathbf{w}^T \mathbf{w} \right) + C \sum_{i=1}^I e_i$$

subjected to the constraint $y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - e_i$ for $I = 1, 2, \ldots, I$ and $e_i \geq 0$.

The equation (33) is a quadratic programming problem and its solution is given by equation (34).

$$f(\mathbf{x}) = \sum \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b$$  \hspace{0.5em} (35)

Now let us consider the case of a non-linear SVM or in other words consider a non-linear separable data set. In such a case we will transform the representation space of the input data into largest dimension space and find the linear classifier in the new space. This is shown in the figure 4-7.
Linear classifier relies on dot product between vectors $x_i^T$ and $x_j$ and it can be represented as shown below:

$$K(x_i, x_j) = x_i^T x_j$$

(36)

Kernel function is the function that corresponds to the inner product in the new high dimensional feature space.

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

(37)

Hence the prediction function represented in equation (35) can be rewritten as shown below.

$$f(x) = \sum \alpha_i y_i K(x_i, x_j) + b$$

(38)

We can have linear, polynomial, Gaussian and sigmoid Kernel functions. SVM handles two classes at a time. For multi class classification we can follow two strategies:

- One versus all
- One versus one

Algorithm for classification using Support Vector Machine can be described in the following steps:

Step 1: Prepare the input to the classifier by converting the input space into feature space.
Step 2: Choose a kernel function and the parameters of the kernel function.

Step 3: Choose the value of C to prevent overfitting.

Step 4: Solve the quadratic programming problem.

Step 5: Construct the discriminant function from the support vectors and find the value of $\alpha_i$.

Step 6: Testing data can now be classified using $\alpha_i$ and the support vectors [39] [38].

In our study we will be using LibSVM. There are many advantages of using the SVM for the purpose of classification. Training of an SVM classifier is very easy and it scales relatively well in the high dimensional data space. The tradeoff between the classifier complexity and the error can be controlled explicitly. This method is very robust in high dimension. The only shortcoming of SVM is that it is sometimes very time consuming depending on the choice of the kernel function.
Chapter 5

Using EMD and ApEn to extract features for classification

5.1 Time series generation

In this study we will generate the continuous phase modulated signal using the Communications System Toolbox for Matlab and Simulink [22]. A model is created in Simulink and the block diagram is as shown in Figure 5-1.

![Figure 5-1. CPM signal generation using Simulink](image)

The Random Integer generator is responsible for the generation of random uniformly distributed integers in a particular range. We need to generate the symbols to feed as an input to the CPM modulator baseband block. In our study we have fixed the M in the M-ary number as 4, hence we can give only 4 symbols as the input to the CPM modulator baseband block and these symbols are +1, -1, +3 and -3. So we have to provide the value of M in the Random Integer generator block and it will generate random uniformly distributed symbols in the range [0, M-1], ie [0,3] in our case since M=4. So as to get only four symbols ±1 and ±3, unipolar to bipolar converter is used. We will be generating 200 symbols for each instance of the class of a particular
value of modulation index. Hence this information must be fed in the Random Integer generator block. In other words, we are providing 200 symbols each of which can be any one of the 4 unique symbols ±1 and ±3 to be modulated. For each symbol we will have 32 samples and this can be done by setting the parameter “Samples per symbol” in CPM modulator baseband to 32. This means that each trial will have a total of 32×200 = 6400 samples. We will be using a rectangular pulse of unit length in our experiment. The output of the CPM modulator baseband block is corrupted with noise to simulate the effects of transmitting a signal through the AWGN channel. This corrupted signal is then saved into a variable and made available in the workspace of Matlab.

5.2 Feature Extraction

Now we have to think about the process of feature extraction. The continuous phase modulated signal with noise is a complex signal. Hence we will first decompose the signal into its real and imaginary parts. Empirical mode decomposition is performed on the real and imaginary part separately to get the intrinsic mode functions. These intrinsic mode functions are subjected to the approximate entropy algorithm to find the degree of irregularity in the signal. IMF 1 has the maximum frequency component and the frequency decreases as the IMF count increases. Hence the ApEn value is relatively greater for the lesser value of IMF count and as the IMF count increases the ApEn value decreases. The ApEn values of each of the IMFs from real and imaginary part of the CPM signal is found and it is used as feature during classification. Apart from these features we are also finding the ApEn values for the real and imaginary part of the signal without decomposing it using EMD. These will also be considered as additional features which is paramount in increasing the classification accuracy. The whole process of feature extraction is shown in Figure 5-2.
5.3 Monte Carlo Simulation

A Monte Carlo simulation is the process of evaluating a deterministic system using a set of random inputs. This is an iterative process and the system is fed with different sets of random inputs. This method is commonly used when a system under evaluation is non-linear, complex or involves a lot of unknown parameters [24]. Another point to be kept in consideration is that the use of random inputs renders a deterministic model of a system into a Stochastic model of the system. Here the input to the system will be uniform random numbers in a particular range.
determined by the maximum and minimum permissible limit. Representation of input in this way introduces an element of uncertainty in this process [23] [24]. Monte Carlo simulation is also known as sampling method because we are sampling a set of random input from the vast population of all the possible sets of random inputs [24].

The steps for uncertainty propagation using Monte Carlo simulation are as follows:

1. Create a parametric model given by the equation: \( y = f(x_1, x_2, \ldots, x_k) \).

2. Generate a set of random uniformly distributed symbols and give it as the input to the model created in step 1. Let the inputs be represented by \( x_{i1}, x_{i2}, \ldots, x_{ik} \).

3. Find the value of \( y_i \) using the equation of the system.

4. To get \( n \) Monte Carlo trials, repeat the step 2 and 3 by getting a new set of random input symbols for \( i=1 \) to \( n \). This is usually done by setting a different value in the initial seed of the random generator every time step 2 is performed [24].

Since we are generating 200 symbols randomly by selecting one of the four symbols \( \pm1 \) and \( \pm3 \) during a single experiment trial, we can consider this as one Monte Carlo trail. In this case the total number of unique sets of random input symbols is \( 4^{200} \). In other words, this is the actual population and we need to sample a set of random input instances from this large pool. In our study we are considering 1000 independent Monte Carlo trials for a single value of modulation index. These sets of instances are created by using a different seed value in the Random Integer generator. In this study the seed value was starts at one and is incremented by unity for a different set of random input symbols. And we have 5 different modulation indices under consideration; hence we have a total of 5000 Monte Carlo trials as a whole.
5.4 Classification Process

The ApEn values of the real and imaginary component of the signal are fed as two features. The IMFs generated by doing the process of EMD on the real part and the imaginary part of the CPM signal is sent through the ApEn algorithm and the corresponding values are fed as the features to the classifier. In our study we considered 12 IMFs each as the output of EMD of the real part and the imaginary part of the signal (This assumption is based on the results of forward feature selection experiment conducted which is mentioned in Section 5.5) and then the ApEn values were found. Hence this accounts to 24 features and additional 2 were added by taking the direct ApEn value of the real and imaginary part of the signal. Hence we have a total of 26 features in a particular instance or input to the classifier. This is represented as F1,F2,…Fn in the Figure 5-3.

A total of 1000 instances are created for each unique value of the modulation index. In this study since we are considering 5 unique values of the modulation index, hence the total number of instances are 5000.

Now we need to train our classifier first. Hence 500 instances are randomly picked from the 1000 instances which are modulated with the same modulation index. This is done for different values of modulation index. A total of 2500 instances are separated and categorized as training data set and the remaining instances are categorized as testing data set. The training data set was used to train our classifier and the misclassification of the testing data was calculated and represented as an error percentage. This is shown in the Figure 5-3. This experiment is conducted many times and the mean value is found for all the error percentages. Determination of Accuracy is done by subtracting the mean error percentage by 100.

\[
Acc = 100 - Error_{mean}
\]
In this study, we will be considering only 12 intrinsic mode functions which are generated by doing the empirical mode decomposition on the real part of the continuous phase modulated signal. The same is the case for the imaginary part of the CPM data signal, we will be only considering 12 intrinsic mode functions. To justify this, I have performed an experiment which uses sequential forward feature selection. Here the goal is feature selection in such a way so as to minimize the misclassification of the classifier or to remove the redundant data so that the computational cost and time can be reduced [25]. In other words sequential forward feature
selection reduces the dimension of the input to the classifier. The forward feature selection algorithm is executed either till the stopping condition is met or till the all the features are used up. In this experiment we have used the error percentage or the misclassification as the selection criterion [25]. Each feature is selected sequentially after a single run of the algorithm and hence the name sequential forward feature selection. This is a bottom-up search method which starts off with an empty feature set F. We will consider each feature and find the error in classification if we only one feature is selected. Find the feature giving the minimum error and add it in the feature set F. Now club all the other features with the feature present in F and form a feature set of 2 features and find the minimum error of classification and add it to the feature set F. Keep on doing the process till we get the minimum misclassification rate. It was found that the lower IMFs had the least amount of information present in them and they were not able to increase the classification accuracy in any way since all the information about the original signal is present in the higher order of the IMFs. Hence dropping the 13th intrinsic mode function won’t affect the classification accuracy.
Chapter 6

Results and Conclusions

In this chapter we will compare the performance of different classifiers. The classifiers under consideration are Fisher’s linear discriminant analysis, Probabilistic neural network and Support vector machine. The optimum value of the c parameter is calculated in each case and the case which gives the best classification for a particular value of c is selected and compared with the classification accuracy of other classifiers.

6.1 Linear Discriminant Analysis is used as the classification method

Here we will find the variation in the accuracy of the classification for different values of SNR when the value of the c parameter used in the ApEn algorithm is fixed to a particular value. The following results are obtained when Linear Discriminant Analysis is used as the classification method. Here each experiment is conducted 10 times for a single value of SNR. The only thing that is different in the 10 experiments is the training data set and the testing data set which collectively has 1000 instances each for a particular value of the modulation index. We will be randomly choosing different sets of data for training and testing from the given 1000 instances. Let’s fix the value of the c parameter. We will also observe the effect of the c parameter on the accuracy of the system. The mean value of the accuracy for the 10 experiments and also the standard deviation is calculated.
Table 6-1. For c parameter = 0.2

<table>
<thead>
<tr>
<th>SNR</th>
<th>mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>81.044</td>
<td>0.515174</td>
</tr>
<tr>
<td>3dB</td>
<td>86.632</td>
<td>0.407834</td>
</tr>
<tr>
<td>5dB</td>
<td>89.72</td>
<td>0.6664</td>
</tr>
<tr>
<td>7dB</td>
<td>90.8</td>
<td>0.936945</td>
</tr>
<tr>
<td>10dB</td>
<td>91.856</td>
<td>1.161179</td>
</tr>
<tr>
<td>13dB</td>
<td>95.372</td>
<td>1.443828</td>
</tr>
<tr>
<td>15dB</td>
<td>96.932</td>
<td>0.470291</td>
</tr>
<tr>
<td>17dB</td>
<td>96.92</td>
<td>0.876559</td>
</tr>
<tr>
<td>20dB</td>
<td>97.16</td>
<td>1.105944</td>
</tr>
</tbody>
</table>

Figure 6-1. Accuracy Verses SNR for c=0.2
Table 6-2. For c parameter = 0.3

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>84.74</td>
<td>0.596955</td>
</tr>
<tr>
<td>3dB</td>
<td>88.68</td>
<td>0.537649</td>
</tr>
<tr>
<td>5dB</td>
<td>90.596</td>
<td>0.393028</td>
</tr>
<tr>
<td>7dB</td>
<td>91.328</td>
<td>1.188807</td>
</tr>
<tr>
<td>10dB</td>
<td>92.44</td>
<td>0.6843</td>
</tr>
<tr>
<td>13dB</td>
<td>95.012</td>
<td>1.819993</td>
</tr>
<tr>
<td>15dB</td>
<td>97.732</td>
<td>0.454528</td>
</tr>
<tr>
<td>17dB</td>
<td>97.968</td>
<td>0.381249</td>
</tr>
<tr>
<td>20dB</td>
<td>98.216</td>
<td>0.563229</td>
</tr>
</tbody>
</table>

Figure 6-2. Accuracy Versus SNR for c=0.3
Table 6-3. For c parameter = 0.4

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>86.792</td>
<td>0.760772</td>
</tr>
<tr>
<td>3dB</td>
<td>89.452</td>
<td>1.694441</td>
</tr>
<tr>
<td>5dB</td>
<td>90.556</td>
<td>0.457413</td>
</tr>
<tr>
<td>7dB</td>
<td>89.468</td>
<td>3.167168</td>
</tr>
<tr>
<td>10dB</td>
<td>92.172</td>
<td>0.891451</td>
</tr>
<tr>
<td>13dB</td>
<td>97.132</td>
<td>0.89959</td>
</tr>
<tr>
<td>15dB</td>
<td>98.216</td>
<td>0.272078</td>
</tr>
<tr>
<td>17dB</td>
<td>98.708</td>
<td>0.287858</td>
</tr>
<tr>
<td>20dB</td>
<td>97.86</td>
<td>0.763442</td>
</tr>
</tbody>
</table>

Figure 6-3. Accuracy Verses SNR for c=0.4
Table 6-4. For c parameter = 0.5

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>87.856</td>
<td>0.870698</td>
</tr>
<tr>
<td>3dB</td>
<td>89.376</td>
<td>1.140635</td>
</tr>
<tr>
<td>5dB</td>
<td>90.48</td>
<td>0.621968</td>
</tr>
<tr>
<td>7dB</td>
<td>90.352</td>
<td>0.922952</td>
</tr>
<tr>
<td>10dB</td>
<td>91.88</td>
<td>0.919517</td>
</tr>
<tr>
<td>13dB</td>
<td>97.708</td>
<td>0.693603</td>
</tr>
<tr>
<td>15dB</td>
<td>98.692</td>
<td>0.355303</td>
</tr>
<tr>
<td>17dB</td>
<td>98.932</td>
<td>0.136039</td>
</tr>
<tr>
<td>20dB</td>
<td>98.204</td>
<td>0.454293</td>
</tr>
</tbody>
</table>

Figure 6-4. Accuracy Verses SNR for c=0.5
Table 6-5. For c parameter = 0.6

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>88.544</td>
<td>0.338992</td>
</tr>
<tr>
<td>3dB</td>
<td>89.852</td>
<td>0.870055</td>
</tr>
<tr>
<td>5dB</td>
<td>90.384</td>
<td>0.969618</td>
</tr>
<tr>
<td>7dB</td>
<td>89.972</td>
<td>2.062667</td>
</tr>
<tr>
<td>10dB</td>
<td>92.68</td>
<td>1.001332</td>
</tr>
<tr>
<td>13dB</td>
<td>98.172</td>
<td>0.600163</td>
</tr>
<tr>
<td>15dB</td>
<td>98.416</td>
<td>1.250522</td>
</tr>
<tr>
<td>17dB</td>
<td>99.204</td>
<td>0.110675</td>
</tr>
<tr>
<td>20dB</td>
<td>97.508</td>
<td>1.035898</td>
</tr>
</tbody>
</table>

Figure 6-5. Accuracy Verses SNR for c=0.6
Here we are comparing the accuracy versus SNR for different values of c parameter so that we can get the best c value to give maximum classification accuracy. We can see that at low value of c we will get a relatively high misclassification rate. As the value of c increases our accuracy increases but if the value of c is too high the variance increases. Hence we can fix the value of c at 0.5 when the Linear Discriminant analysis is used for classification.

6.2 Probabilistic Neural Network is used as the classification method

Now let’s consider using PNN for finding the accuracy versus SNR for different values of the c parameter. Our goal is to find the best value of c having highest accuracy with a limited variance in the output of the 10 experiments.
Table 6-6. For c parameter = 0.2

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>74.376</td>
<td>1.325738</td>
</tr>
<tr>
<td>3dB</td>
<td>82.26</td>
<td>1.683092</td>
</tr>
<tr>
<td>5dB</td>
<td>84.016</td>
<td>1.032486</td>
</tr>
<tr>
<td>7dB</td>
<td>87.244</td>
<td>0.447988</td>
</tr>
<tr>
<td>10dB</td>
<td>87.068</td>
<td>0.922807</td>
</tr>
<tr>
<td>13dB</td>
<td>86.408</td>
<td>0.380783</td>
</tr>
<tr>
<td>15dB</td>
<td>89.536</td>
<td>0.66672</td>
</tr>
<tr>
<td>17dB</td>
<td>88.252</td>
<td>0.341005</td>
</tr>
<tr>
<td>20dB</td>
<td>91.4</td>
<td>0.637216</td>
</tr>
</tbody>
</table>

Figure 6-7. Accuracy Verses SNR for c=0.2
Table 6-7. For c parameter = 0.3

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>78.488</td>
<td>1.585692</td>
</tr>
<tr>
<td>3dB</td>
<td>82.396</td>
<td>0.771365</td>
</tr>
<tr>
<td>5dB</td>
<td>79.212</td>
<td>1.105148</td>
</tr>
<tr>
<td>7dB</td>
<td>86.572</td>
<td>0.774952</td>
</tr>
<tr>
<td>10dB</td>
<td>85.028</td>
<td>0.741662</td>
</tr>
<tr>
<td>13dB</td>
<td>85.848</td>
<td>0.30217</td>
</tr>
<tr>
<td>15dB</td>
<td>88.704</td>
<td>0.430896</td>
</tr>
<tr>
<td>17dB</td>
<td>87.232</td>
<td>0.393186</td>
</tr>
<tr>
<td>20dB</td>
<td>92.496</td>
<td>0.406699</td>
</tr>
</tbody>
</table>

Figure 6-8. Accuracy Verses SNR for c=0.3
Table 6-8. For c parameter = 0.4

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>76.128</td>
<td>0.975213</td>
</tr>
<tr>
<td>3dB</td>
<td>80.76</td>
<td>0.69946</td>
</tr>
<tr>
<td>5dB</td>
<td>75.72</td>
<td>0.868409</td>
</tr>
<tr>
<td>7dB</td>
<td>85.368</td>
<td>0.834756</td>
</tr>
<tr>
<td>10dB</td>
<td>80.684</td>
<td>0.519641</td>
</tr>
<tr>
<td>13dB</td>
<td>81.944</td>
<td>0.521604</td>
</tr>
<tr>
<td>15dB</td>
<td>87.504</td>
<td>0.429656</td>
</tr>
<tr>
<td>17dB</td>
<td>85.18</td>
<td>0.540041</td>
</tr>
<tr>
<td>20dB</td>
<td>92.172</td>
<td>0.273285</td>
</tr>
</tbody>
</table>

Figure 6-9. Accuracy Verses SNR for c=0.4
Table 6-9. For c parameter = 0.5

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>76.304</td>
<td>1.010206</td>
</tr>
<tr>
<td>3dB</td>
<td>78.68</td>
<td>0.561585</td>
</tr>
<tr>
<td>5dB</td>
<td>73.26</td>
<td>0.612536</td>
</tr>
<tr>
<td>7dB</td>
<td>84.024</td>
<td>0.455368</td>
</tr>
<tr>
<td>10dB</td>
<td>76.416</td>
<td>0.749566</td>
</tr>
<tr>
<td>13dB</td>
<td>77.956</td>
<td>0.90793</td>
</tr>
<tr>
<td>15dB</td>
<td>85.844</td>
<td>0.567396</td>
</tr>
<tr>
<td>17dB</td>
<td>82.504</td>
<td>0.30533</td>
</tr>
<tr>
<td>20dB</td>
<td>87.676</td>
<td>0.919024</td>
</tr>
</tbody>
</table>

Figure 6-10. Accuracy Versus SNR for c=0.5
Table 6-10. For c parameter = 0.6

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>78.964</td>
<td>0.788461</td>
</tr>
<tr>
<td>3dB</td>
<td>76.192</td>
<td>0.538904</td>
</tr>
<tr>
<td>5dB</td>
<td>71.488</td>
<td>0.789667</td>
</tr>
<tr>
<td>7dB</td>
<td>80.04</td>
<td>0.676199</td>
</tr>
<tr>
<td>10dB</td>
<td>73.708</td>
<td>1.492148</td>
</tr>
<tr>
<td>13dB</td>
<td>74.908</td>
<td>0.571952</td>
</tr>
<tr>
<td>15dB</td>
<td>83.74</td>
<td>0.850464</td>
</tr>
<tr>
<td>17dB</td>
<td>79.824</td>
<td>0.564805</td>
</tr>
<tr>
<td>20dB</td>
<td>82.224</td>
<td>1.442615</td>
</tr>
</tbody>
</table>

Figure 6-11. Accuracy Verses SNR for c=0.6
When a Probabilistic neural network is used for the purpose of classification, we can observe from the plots in this section that the accuracy of classification is high for low values of c. Hence we can consider 0.2 or 0.3 as a good selection for the value of c. But it can be concluded that the LDA performs far better than the probabilistic neural network for this kind of data set. We will also compare the performance of PNN against LDA and SVM in section 6.4.

6.3 Support Vector Machine is used as the classification method

Now let’s use the SVM for classification of the instances to their corresponding classes. The experiment is done 10 times and the mean and standard deviation of accuracy is calculated.
Table 6-11. For c parameter = 0.2

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>81.112</td>
<td>0.476953</td>
</tr>
<tr>
<td>3dB</td>
<td>87.44</td>
<td>0.748806</td>
</tr>
<tr>
<td>5dB</td>
<td>90.228</td>
<td>0.602528</td>
</tr>
<tr>
<td>7dB</td>
<td>91.952</td>
<td>0.458713</td>
</tr>
<tr>
<td>10dB</td>
<td>92.916</td>
<td>1.151995</td>
</tr>
<tr>
<td>13dB</td>
<td>94.996</td>
<td>0.307217</td>
</tr>
<tr>
<td>15dB</td>
<td>96.352</td>
<td>0.28019</td>
</tr>
<tr>
<td>17dB</td>
<td>95.788</td>
<td>0.518133</td>
</tr>
<tr>
<td>20dB</td>
<td>97.54</td>
<td>0.19799</td>
</tr>
</tbody>
</table>

Figure 6-12. Accuracy Versus SNR for c=0.2
Table 6-12. For c parameter = 0.3

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>80.724</td>
<td>4.384881</td>
</tr>
<tr>
<td>3dB</td>
<td>88.52</td>
<td>1.466424</td>
</tr>
<tr>
<td>5dB</td>
<td>91.012</td>
<td>0.625403</td>
</tr>
<tr>
<td>7dB</td>
<td>91.12</td>
<td>0.96646</td>
</tr>
<tr>
<td>10dB</td>
<td>94.506</td>
<td>0.35877</td>
</tr>
<tr>
<td>13dB</td>
<td>95.988</td>
<td>0.194182</td>
</tr>
<tr>
<td>15dB</td>
<td>96.788</td>
<td>0.256506</td>
</tr>
<tr>
<td>17dB</td>
<td>96.64</td>
<td>0.392372</td>
</tr>
<tr>
<td>20dB</td>
<td>97.876</td>
<td>0.096977</td>
</tr>
</tbody>
</table>

Figure 6-13. Accuracy Verses SNR for c=0.3
Table 6-13. For c parameter = 0.4

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>82.372</td>
<td>5.940716</td>
</tr>
<tr>
<td>3dB</td>
<td>90.508</td>
<td>0.527316</td>
</tr>
<tr>
<td>5dB</td>
<td>91.16</td>
<td>0.371902</td>
</tr>
<tr>
<td>7dB</td>
<td>92.208</td>
<td>0.589327</td>
</tr>
<tr>
<td>10dB</td>
<td>94.745</td>
<td>0.37598</td>
</tr>
<tr>
<td>13dB</td>
<td>96.54</td>
<td>0.147573</td>
</tr>
<tr>
<td>15dB</td>
<td>97.383</td>
<td>0.198553</td>
</tr>
<tr>
<td>17dB</td>
<td>97.396</td>
<td>0.219454</td>
</tr>
<tr>
<td>20dB</td>
<td>98.012</td>
<td>0.170802</td>
</tr>
</tbody>
</table>

Figure 6-14. Accuracy Verses SNR for c=0.4
Table 6-14. For c parameter = 0.5

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>85.42</td>
<td>3.227397</td>
</tr>
<tr>
<td>3dB</td>
<td>90.912</td>
<td>0.746857</td>
</tr>
<tr>
<td>5dB</td>
<td>90.884</td>
<td>1.053325</td>
</tr>
<tr>
<td>7dB</td>
<td>91.52</td>
<td>1.193315</td>
</tr>
<tr>
<td>10dB</td>
<td>95.276</td>
<td>0.2996</td>
</tr>
<tr>
<td>13dB</td>
<td>97.172</td>
<td>0.230208</td>
</tr>
<tr>
<td>15dB</td>
<td>97.928</td>
<td>0.127611</td>
</tr>
<tr>
<td>17dB</td>
<td>97.776</td>
<td>0.253824</td>
</tr>
<tr>
<td>20dB</td>
<td>98.228</td>
<td>0.126561</td>
</tr>
</tbody>
</table>

Figure 6-15. Accuracy Verses SNR for c=0.5
Table 6-15. For c parameter = 0.6

<table>
<thead>
<tr>
<th>SNR</th>
<th>Mean of Accuracy values</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1dB</td>
<td>85.836</td>
<td>2.377035</td>
</tr>
<tr>
<td>3dB</td>
<td>91.464</td>
<td>0.696455</td>
</tr>
<tr>
<td>5dB</td>
<td>90.22</td>
<td>0.871321</td>
</tr>
<tr>
<td>7dB</td>
<td>93.032</td>
<td>0.334026</td>
</tr>
<tr>
<td>10dB</td>
<td>95.716</td>
<td>0.259538</td>
</tr>
<tr>
<td>13dB</td>
<td>97.436</td>
<td>0.303725</td>
</tr>
<tr>
<td>15dB</td>
<td>98.248</td>
<td>0.14703</td>
</tr>
<tr>
<td>17dB</td>
<td>96.94</td>
<td>0.785253</td>
</tr>
<tr>
<td>20dB</td>
<td>98.496</td>
<td>0.113451</td>
</tr>
</tbody>
</table>

Figure 6-16. Accuracy Verses SNR for c=0.6
By analyzing the column graph with the error bars we can observe that the accuracy of classification increases as the value of c parameter increases. We can safely say that the value of c as 0.5 or 0.6 will be good for minimizing the misclassification. It is also observed that the standard deviation of the accuracy percentage is very high for low values of SNR. Hence the classification system using support vector machine should be avoided at low values of SNR. This issue hampers the robustness of the system.

6.4 Comparison between the different classification systems (LDA, PNN and SVM)

In this section we will be comparing different classification system and find which system has the maximum classification accuracy. The robustness factor will also be taken into consideration. However it should be kept in mind that we will not be trying to generalize the result obtained here for all the data sets. The comparison between the classifiers is done...
completely based on the experimental results and is applicable only to the data sample that was obtained after empirical mode decomposition of a CPM signal and then finding the approximate entropy of each of the intrinsic mode functions generated by the EMD process. From section 6.1, we had inferred that maximum classification accuracy is obtained for the value of the c parameter as 0.5 when linear discriminant analysis method is used for classification. The case when a probabilistic neural network is used for the classification purpose is shown in great detail in section 6.2. We had concluded that the value of c parameter as 0.2 or 0.3 will optimize our classifier. Taking the standard deviation into account we can safely fix the value of c at 0.3 in the case of PNN. Now we take the SVM into consideration and we can fix the value of c at 0.5 and this is shown graphically in section 6.3.

Figure 6-18. Accuracy Versus SNR when different Classification techniques are used
Now let us draw inference from the column plot with error bars shown in Figure 6-18. We can clearly see that classification using linear discriminant analysis and support vector machine have much greater classification accuracy when compared with the method of classification using probabilistic neural network. We can also see that a classifier using LDA performs better than the classifier using support vectors. Although the classification accuracy of the SVM classifier is comparable to the classifier using LDA in the mid ranges of SNR but the standard deviation of the data collected from 10 experiments at low value of SNR (1dB) is very high and hence this creates a very less robust classification system. Hence classification using linear discriminant analysis comes out as a good option for this data set.
Appendix

Estimation of SNR

Estimating the value of the SNR of a signal is very important in many applications which take the estimated value of SNR as a priori information. The algorithm which is used to determine the SNR value is given in [40] and [41]. This algorithm estimates the value of SNR based on iterative subspace tracking for digitally modulated signals. Let the signal which is received be represented by \( r(t) \) and the signal that was transmitted be represented as \( x(t) \) and the additive white Gaussian noise is represented by \( n(t) \). Hence the received signal can be written as

\[
\begin{equation}
\begin{aligned}
\mathbf{r}(t) = \mathbf{x}(t) + \mathbf{n}(t)
\end{aligned}
\end{equation}
\]

(1)

The sampled form of the above equation is given by equation (2).

\[
\begin{equation}
\begin{aligned}
\mathbf{r}_k = \mathbf{x}_k + \mathbf{n}_k
\end{aligned}
\end{equation}
\]

(2)

Here \( \mathbf{x}_k \) is the sample of a signal which is digitally modulated using a carrier which has a frequency at \( f_c \) and \( \mathbf{n}_k \) is the AWGN sample with zero mean and a variance of \( \sigma_n^2 \).

Let us find the correction matrix of the received signal \( \mathbf{r}(t) \) where

\[
\begin{equation}
\begin{aligned}
\mathbf{r}(t) = [\mathbf{r}_t, \mathbf{r}_{t-1}, ..., \mathbf{r}_{t-L+1}]^T
\end{aligned}
\end{equation}
\]

It is been assumed that we have \( L \) samples at time instant \( t \).

The correlation matrix is denoted by \( \mathbf{R}_r \) and \( \mathbf{R}_r = E[\mathbf{r}(t)\mathbf{r}^H(t)] \)

This correlation matrix can also be expressed as the sum of the correlation matrix of the digitally modulated signals and the correlation matrix of the AWGN. This is shown in equation (4).

\[
\begin{equation}
\begin{aligned}
\mathbf{R}_r = \mathbf{R}_x + \mathbf{R}_n
\end{aligned}
\end{equation}
\]

(4)

Now we will do eigenvalue decomposition of the covariance matrix \( \mathbf{R}_r \) which can be represented as \( \mathbf{R}_r = \mathbf{A}\mathbf{D}\mathbf{A}^H \) where \( \mathbf{A} \) is the matrix which is formed by orthonormal eigenvectors and \( \mathbf{D} \) is a diagonal matrix which contains the corresponding eigenvalues in the descending order. Let the eigenvalues be represented in the descending order as \( b_1 \geq b_2 \geq \cdots \geq b_L \).
The correlation matrix for the noise is given by equation (5).

\[ R_n = \sigma_n^2 I \] (5)

The eigenvalues of the correlation matrix \( R_f \) can be written as follows due to the relation mentioned in equation (4).

\[ b_i = \begin{cases} \sigma_f^2 + \sigma_n^2 & 1 \leq i \leq p \\ \sigma_n^2 & p + 1 \leq i \leq L \end{cases} \] (6)

We can infer from equation (6) that the signal subspace is defined by the first \( p \) eigenvectors and the rest \( L-p \) eigenvectors are only influenced by the noise subspace. It is important to find the dimension of the signal subspace which is denoted by \( p \). This dimension can be found easily using the Minimum description length function [40] which is defined in equation (7).

\[
MDL(k) = -log \left( \frac{\prod_{i=k+1}^{L} b_i^{\frac{1}{(L-k)}}}{1 \sum_{i=k+1}^{L} b_i} \right)^{(L-k)N} + \frac{1}{2} k(2L - k) log N
\] (7)

The dimension of the signal space is the value of \( k \) which minimizes equation (7). This is shown in equation (8).

\[ p = \arg \min_k MDL(k) \text{ where } k = \{0,1,2,...,L-1\} \] (8)

Since the actual value of the correlation matrix is very difficult to find due to the limited number of received data, we will find an estimate of the matrix as shown in equation (9).

\[ R_{est} = \frac{1}{K} \sum_{k=1}^{K} r(k)r^H(k) \] (9)

Now let’s summarize the algorithm for estimating the value of SNR. The steps for the same are as shown below:

Step 1: Compute the correlation matrix of the received signal

Step 2: Find the eigenvalues and arrange it in descending order.

Step 3: Find the dimension of the signal subspace \( p \) from equation (7)

Step 4: Find the noise power using the equation (10)
\[ P_N = \sigma_n^2 = \frac{1}{L - p} \sum_{i=p+1}^{L} b_i \]  \hspace{1cm} (10)

Step 5: The signal power can be estimated by using the following equation.

\[ P_s = \sum_{i=1}^{p} (b_i - \sigma_n^2) \]  \hspace{1cm} (11)

Step 6: We can find the SNR using equation (10) and (11) and put it in equation (12).

\[ SNR = 10 \log \left( \frac{P_s}{P_N} \right) \]  \hspace{1cm} (12)
REFERENCES


[21] Christopher M. Bishop “Pattern Recognition and Machine Learning”

[22] The MathWorks, Inc. “Communications Blockset For Use with Simulink”


[28] Colin Campbell and Yiming Ying, “Learning with Support Vector Machines” SYNTHESIS LECTURES ON ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING #10, Morgan and Claypool publishers


[38] Prof. Takagi and Nakai “SVM, kernel methods and bioinformatics”, Human genome center, University of Tokyo, Tokyo, Japan


