THE UNIVERSITY
of LIVERPOOL

AUTOMATIC CLASSIFICATION OF DIGITAL COMMUNICATION MODULATION SCHEMES

A thesis submitted to the University of Liverpool
for the degree of Doctor of Philosophy
in the Faculty of Engineering

September 2003

By

Dennis Mou Ling Wong
Department of Electrical Engineering and Electronics

Contents

List of Tables

List of Figures

Abstract

Digitally modulated signals play an important role in the world of communications today. It is increasingly necessary for the communication intelligence (COMINT) sector to have the capability to classify these signals effectively in an automated fashion. Automatic recognition of a signal's modulation has been a well studied subject and has undergone significant transformation in recent years.

This thesis investigates some recent advances in the field of automatic modulation classification, especially in the development of modulation classifiers. From the view of organisation, this thesis can be largely broken into three main parts, i.e.:

i) investigation of neural networks (NN) approach

ii) refinement of NN approach through feature selection; and

iii) investigation of maximum likelihood (ML) approach.

For the NN approach, a conventional multi-layer perceptron (MLP) based classifier is reviewed and refined in performance through the adoption of a more robust algorithm. Furthermore, a wider variety of NNs are also investigated including radial basis function (RBF) network and probabilistic neural network (PNN). Additionally, A novel feature set based on the analysis of higher-order statistics of the signal's constellation is also formulated in this work. From experiments, the proposed feature set has shown a good degree of robustness and invariance under additive white and Gaussian noise (AWGN) environments.

A good feature set is essential for a NN classifier to achieve good performance. However, as the number of features increases, the number of training examples needed to ensure statistically reliable performance also increases. This presents a practical problem. The second part of the thesis investigates the possibility of minimising the number of input features with the minimum degradation in classification performance. New genetic algorithm (GA) based methods are devised for the selection of features for modulation classification. The rest of this part of the work is devoted to the investigation of linear transformation based methods, in particular, principal component analysis (PCA) and independent component analysis (ICA).

When training examples are not available, one has to resort to alternative approaches that do not require training examples. In the final part of this work, ML based methods are reviewed and investigated. In a coherent environment ML classifier provides an upper bound for classification performance. A new minimum distance based method is derived from ML method and its performance is compared with the ML method. Although this method suffers degradation of 2 dB in performance (in a dual class scenario), it offers better computational efficiency. A multi-class scenario is also investigated here.

Although ML is the optimum classifier in a coherent environment, its performance is expected to suffer from degradation in non-coherent environment as phase mismatch is introduced. Therefore, a closed form ICA based algorithm is introduced to mitigate the effects of phase errors. Originally devised for blind source separation (BSS), the algorithm is proven capable of removing the phase offset through experiments.

In short, this thesis is comprised of a collection of recent works in automatic modulation classification. When training samples are available, NN classifiers provides a robust and effective solution. Feature selection and transformation are efficient tools to form a smaller feature subset for NN classifier with minimum compromise in performance. On the other hand, when training examples are not available ML classifiers provide an optimum solution in a coherent environment. With the aid of a closed form phase offset removal blockset, the ML classifier can be applied in the non-coherent operating environment directly.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institution of learning.

Copyright

Acknowledgements

天行健，君子以自强而不息。

–易经

There are many people whom I am greatly in debt to; for their helps throughout the duration of my Ph.D. in The University of Liverpool. First and foremost, my gratitude must go to my Ph.D. supervisor, Prof. Asoke K. Nandi, who has been constantly helping and guiding me throughout this period with matters within and outside of academic progression. I must also thank the Overseas Research Studentship (ORS), UK and The University of Liverpool for funding this research. A note of appreciation must also be made to Dr. Elsayed E. Azzouz for his correspondence at the initial phase of this research.

I couldn't have possibly get started in my research and finished the thesis without my team of colleagues from the Signal Processing and Communications Group, of whom some have left before me. In particular, I can't get away if I do not express my thanks to Dr. Lindsay Jack, Dr. Vicente Zarzoso, Dr. Frank Herrmann[1] and Mr. Andrew Parkins. Thank you for your many enlightening and insightful discussions and practical supports. To all colleagues, thanks for your tolerance and amazing pure Liverpudian sense of humour.

The Department of Electrical Engineering and Electronics has been a great place to do my undergraduate studies and my postgraduate studies. My sincere

[1]now in Robert Gordon University, Scotland

gratitude's to those who had in one way or another made my duration here a, dare I say, "electric" one! Not forgetting Mr. David Turner, Mrs. Brenda Lussey, Mrs. Angela Winker, Miss Jenny Mullis and Miss Alison Reid for their numerous assistance in administrative matters.

My entire congregation of Liverpool Chinese Gospel Church has made my stay in Liverpool an enjoyable and unforgettable experience. I wish to thank each and every one of you for the many meals and fellowship we share. Special appreciation must be made to Ivy, my housemate, without whom this thesis could not be finished on schedule.

Lastly, thanks must also be made to my parents who encouraged me to start my research career in the first place, and Sandy, for her patience and encouragement throughout these years.

Chapter 1

Introduction

Communication is an integral part of our life nowadays. We often encounter communication without realising the process of it, for examples, the telephones (both land-lines and mobile phones) we use every day, the computer terminal, radio and television networks, and the Internet are examples of many facets of communications that surround us today. Many of these applications are multiplexed in nature, i.e. many sources of signals share a common channel. One way of multiplexing these signals is to limit the frequency bandwidth of the signal and send the signals at different portions of the total available bandwidth. To do this, we require an important process in communication called modulation. In its simplest form, modulation is the process where the carrier wave (often at a higher frequency band) is "modulated" by the information that is to be transmitted. For digital modulation, this process is preceded by an analogue-to-digital conversion (ADC) phase; and a symbol-mapping phase whereby the digitised signal is mapped onto the respective constellation diagrams.

Today, digitally modulated communication signals are becoming vital as the world is becoming increasingly 'digital'. Many analogue applications have been replaced with their digital counterparts e.g. digital cable/satellite television, digital audio and digital software radio. The wide usage of Internet is another evidence of the popularity of digital signals, digital modems are used to (de)modulate data for transmitting across normal telephone lines. Additionally, the mobile communication system has been 'digitised' with the introduction of GSM networks around the globe and is now getting a further boost with the introduction of third generation (3G) code division multiple access (CDMA) based system [1, 2, 3, 4]. The forthcoming fourth generation mobile system (4G) [5, 6] has also proposed an IP-based digital network with the adoption of multi-carrier techniques. With the shift of focus from analogue technology to digital equivalents, digital modulation now deserves more attention than the aging analogue modulation.

Software defined radio (SDR) [7, 8, 9] provides an interesting prospect for the future communications sector, as it has the ability of rapid reconfiguration and as all processes in SDR are executed in software, it is also cheaper to manufacture hence beneficial financially. Another advantage of SDR is that, it provides a single unified platform for signals of various modulation types. This highly desirable feature implies that we need an 'intelligent' modem that can automatically identify different modulation types. Automatic recognition of the modulation type of the incoming signal then plays an important task in this 'intelligent' modem.

In this work, we investigate several variants of such study, under the umbrella of a wider field of pattern recognition. In digital communication, each modulation has its own distinct characteristics that can be considered as an unique pattern. The aim of this study is to seek an efficient method for classifying these patterns with very limited amount of prior knowledge. In the originating work, Liedtke [10] described modulation classification is neither an energy detection nor a demodulation problem that required full message extraction, but it is something in between as depicted in Figure 1.1. The amount of prior knowledge required for modulation recognition is more than what is needed for symbol detection, and

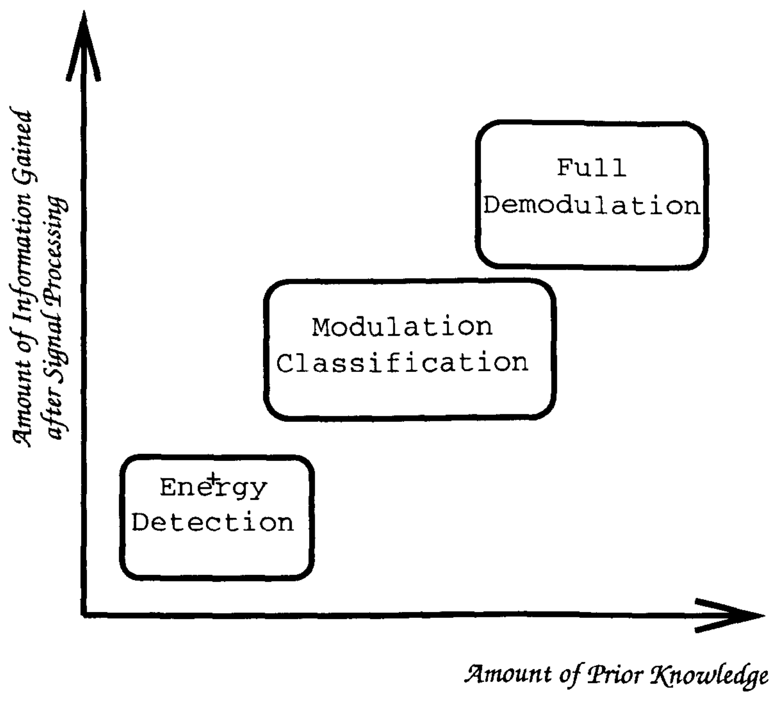

Amount of Information Gained after Signal Processing
Full Demodulation
Modulation Classification
Energy Detection
Amount of Prior Knowledge

Figure 1.1: The relationship of ADMC and other detection procedures

similar to that required for demodulation. However, due to the nature of the problem, we do not possess these information and estimates have be done blindly without the actual knowledge. Correct classification would not give us the full demodulated information, and more complex tasks of deciphering might follow after demodulation. Nevertheless, classification of modulation type is an essential process especially in a furtive environment.

In recent years, artificial intelligence (AI) has found many applications in pattern recognition and established a good track record in performance against traditional statistical pattern recognition. The use of AI is also found in various complex applications among respective engineering disciplines. For example, Artificial Neural Networks (ANN) have been applied in system identification [11], and system control processes [12]; Genetic Algorithms (GA) in optimisation process [13, 14], multiuser detectors for DS-CDMA communications [15], and Fuzzy logic in robotics control [16], etc.

By framing the task of modulation classification into a supervised neural network learning problem, we avoid the cumbersome process of manual determination of thresholds for key features (Chapter 3) as in conventional decision theoretic methods (see Chapter 2). Besides, through this formulation using ANNs, one also benefits from the many advantages offered by ANN detailed in Chapter 4.

There are limitations for any supervised learning method. One of these is the input dimension of a learning machine. The larger the input dimension of the learning machine is, the larger the amount of training data that is required for drawing a sufficiently generalised discriminating boundary. This problem in itself is an interesting one and is to be investigated in an individual study in its own right. Such study is sometimes named feature selection (see Chapter 5). The objective of feature selection is to select a minimal amount of input dimension while avoiding performance loss. At times, the outcome of feature selection can increase the performance of the learning machine.

Another limiting factor for the supervised learning machine would be the need for training data. At times, there is too much variability in a system, and it is desirable to have a generalised framework for all variabilities, e.g. changes in signal to noise ratio, fading channels, etc. The amount of training examples needed to cover such a wide variety of scenario is just not feasible in practice. Therefore, it would be logical to re-framework the existing problem into a probabilistic hypothesis testing one. However, unlike conventional decision theoretic methods, we would like to consider one specific form of classifier that would handle multiple hypotheses at once, so that the order of tests does not have any influence on the final classification result. One way of doing so is to have a likelihood value for each possible hypothesis and choose the hypothesis with the greatest likelihood value. This is called the maximum likelihood (ML) method and will be discussed in greater depth later (Chapter 6).

We cannot overlook the explicit usage of probability theory and higher order statistics (HOS) in this work. Gaussian probability density function (PDF) can be completely characterised using second order statistics (SOS), i.e. by the mean and variance of the distribution. However, real world data often tends to be non-Gaussian, and in this case, SOS cannot provide sufficient information to help to solve the problem. Hence, HOS provides an interesting approach to many problems, e.g. independent component analysis (ICA) and blind source separation (BSS), blind channel equalisation, blind SNR estimation, blind deconvolution, etc. These applications are termed 'blind' because one does not have the prior knowledge on the target's information, e.g. different components in ICA or channel coefficient in the case of channel estimation or deconvolution. HOS plays an integral role in this work, especially in feature formulation and extraction. Besides, HOS is also employed in non-coherent environment prior to the classification.

1.1 Structural Organisation of The Thesis

This thesis is organised in the following fashion: Chapter 2 presents an introductory section on the background and motivations of this research; this is followed by another section introducing some communications and mathematical preliminaries which might be useful for readers that do not have necessary pre-requisites toward this study. Next, Chapter 3 will present the signal model used in this work and subsequently the derivation of a novel set of HOS based features. This feature set is in turn used in the experimental works of Chapter 4, preceded by an introduction to ANN, and some feed-forward variants of the ANN.

Chapter 5 expands the feature set to include a wider range of statistical and spectral features to investigate the interesting topic of feature selection. Two paradigms are discussed in this chapter: choosing feature subset by selection and forming feature subset by transformation. Both paradigms were supported with a set of results obtained via computer experiments.

Furthermore, in the final part of the thesis (Chapter 6), the ML likelihood classifier and its variants are introduced and investigated. The classifiers are first derived in the ideal condition, where the signals are only corrupted through an AWGN channel. Later on, the investigation was taken further in an non-coherent environment where a ICA/BSS based method is called upon to mitigated the phase offset introduced in the received signal model as a result of carrier frequency offset.

Lastly, the work is summarised in Chapter 7 on a chapter-by-chapter basis. A summary of possible directions for the future works of this research is also presented at the end of this chapter.

1.2 Summary of Contributions

This thesis has some novel works in the study of automatic digital modulation classification and these contributions are summarised as below:

- A set of new higher order cumulants based features had been devised and analysed in Chapter 3. These features (a set of three 4^{th} order cross cumulants) are guaranteed to be non-complex. Besides, the standard deviations of these features are shown to be small under the influence of addictive white and Gaussian noise.

- An alternate training algorithm, the Resilient Backpropagation (RPRPOP), were proposed in favour to the conventional backpropagation (BP) algorithm adopted in the previous works. Expriemental results showed that RPROP is faster to train and offers better classification performance. The above is detailed in Chapter 4.

- The author proposed a new integer list genome based genetic algorithm (GA), in Chapter 5. This integer list genome mitigate the posibility of repeatable entry into the GA, hence reducing the search space. Unlike the binary genome based GA, the new GA gives the users control over the number of required features.

- A comparison of principal component analysis (PCA) and independent component analysis (ICA) as feature transformation techniques has been made in second part of Chapter 5.

- In Chapter 6, a new maximum likelihood (ML) modulation classifier (MC) with SNR estimation had been devised and introduced. The new classifier showed high degree of robustness in coherent AWGN environment. Its performance had been compared to the optimum ML classifier with *a priori* knowledge of SNR.

- Besides, the author employed a Blind Source Separation (BSS) algorithm in non-coherent AWGN environment. By doing so, one can directly employ the above mentioned ML MC without alteration to the original algorithm. The performance of such algorithms had been investigated through experimental results.

1.3 List of Publications

This research started in September 1999 and was the major part of the author's research. However, there are also some other works, both in communication and pattern recognition, that the author has undertaken through this period of time. All these are reflected in the following list of publications:

- **M. L. D. Wong** and A. K. Nandi, "Multiuser Detection", Proceedings of ICCCD'2000, Khagapur,India, December 2000.
- **M. L. D. Wong** and A. K. Nandi, "Automatic Digital Modulation Recognition using Spectral and Statistical Features with Multi-layer perceptrons", Proceedings of ISSPA 2001, pp. 390-393, Vol II, ISSPA, Kuala Lumpur, Malaysia, August 2001.
- L. B. Jack, **M. L. D. Wong** and A. K. Nandi, "Modified Kohonen Self Organizing Map for Automated Fault Detection in Helicopter Gear Boxes", Proceedings of the 16th Int. Congress and Exhibition on Condition Monitoring and Diagnostic Engineering Management (COMADEM), Sweden, 27-29 August 2003.
- **M. L. D. Wong** and A. K. Nandi, "Automatic Digital Modulation Recognition using Artificial Neural Network and Genetic Algorithm", Signal Processing 84(2)(2004), 351-365.
- **M. L. D. Wong**, L. B. Jack and A. K. Nandi, "Automated Novelty Detection using a modified Kohonen Self Organising Map", preprint submitted to EURASIP Journal on Applied Signal Processing.
- L. B. Jack, **M. L. D. Wong** and A. K. Nandi, "A Modified SOM for Helicopter Gear Box Fault Detection", presented at Sensors for Aircraft Systems, Institute of Physics. 14 May 2003.

Chapter 2

Background and Preliminaries

In this chapter, a section on the background of automatic modulation classification and the related motivationsis introduced. The background is taken from a historical viewpoint and ends with discussions on some of the recent developments of this study. The second part of this chapter is devoted to some pre-requisites for readers who may not have the necessary academic background.

2.1 Background and Motivations

The present methods of modulation recognition can be traced back to Liedtke's first paper [10] published in 1984. Since then, the methods became very different from what was first proposed. Azzouz and Nandi's pioneering works in the 90's led to their book [17]; this presented a excellent summary of recent modulation recognition techniques up till 1996. They proposed various algorithms in dealing with analogue modulation and digital modulation signals. Besides, they also proposed the use of ANN in complement with conventional hypothesis testing classification methods. Various new features were introduced for differentiating among various classes of analogue and digital modulations.

In the early days, modulation recognition relied heavily on the human operator's interpretation of measured parameters to classify signals. Signal properties such as IF waveform, signal spectrum, instantaneous amplitudes and instantaneous phase are often used in conventional methods. A latter form of recogniser (Figure 2.1) consists of a bank of demodulators, each designed for a particular modulation type. This is considered as semi-automatic since a human operator is still required to 'listen' to the output, but it is impractical for digital communications.

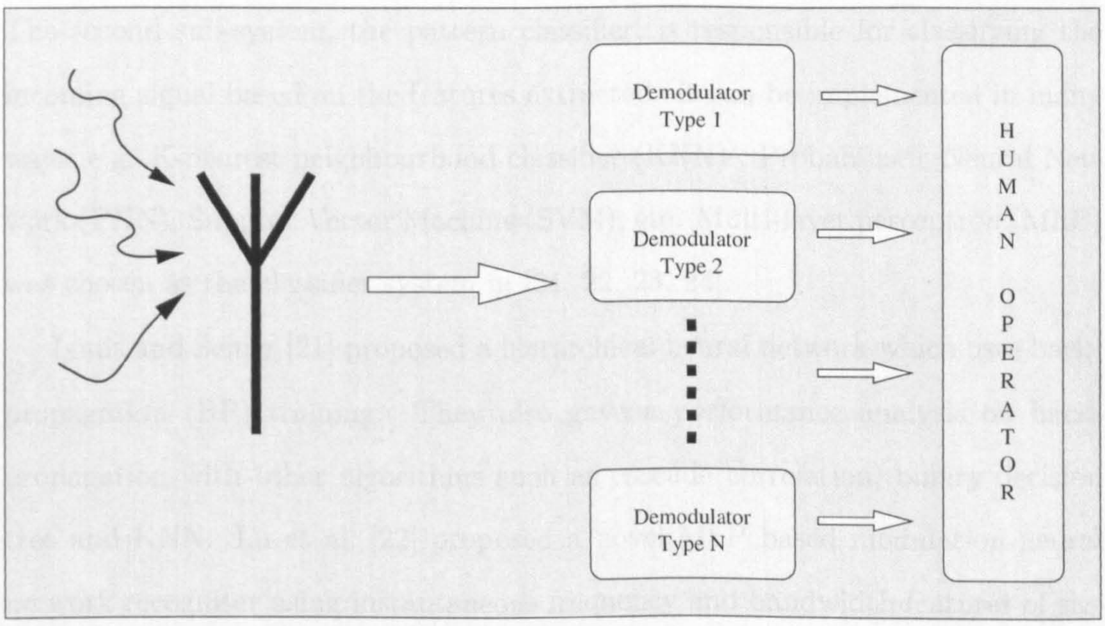


Figure 2.1: A semi-automatic modulation classifier which requires human interpretation.

Since the mid-80's, new classes of modulation recognisers which automatically determine incoming modulation type have been proposed. Generally, these methods fall into two main categories, decision theoretic and statistical pattern recognition. Decision theoretic approaches use probabilistic and hypothesis testing arguments to formulate the recognition problem. The major drawback of this approach are the difficulties of forming the right hypothesis as well as careful analyses that are required to set the correct threshold values. Examples of decision theoretic approaches include Azzouz and Nandi [18, 19] who proposed a global procedure for analogue and digital modulation signals; and Dohono and Huo [20] who proposed a new method using Hellinger representation.

Pattern recognition approaches, however, do not need such careful treatment. Nevertheless choosing the right feature set is still an important issue. The recognition method can be further divided into two: the feature extraction sub-system and the classification sub-system. The feature extraction sub-system is responsible for extracting prominent characteristics, called features, from the raw data. The second sub-system, the pattern classifier, is responsible for classifying the incoming signal based on the features extracted. It can be implemented in many ways, e.g. K-nearest neighbourhood classifier (KNN) , Probabilistic Neural Network (PNN), Support Vector Machine (SVM), etc. Multi-layer perceptron (MLP) was chosen as the classifier system in [21, 22, 23, 24].

Louis and Sehier [21] proposed a hierarchical neural network which uses back-propagation (BP) training. They also gave a performance analysis on back-propagation with other algorithms such as cascade correlation, binary decision tree and KNN. Lu et al. [22] proposed a novel MLP based modulation neural network recogniser using instantaneous frequency and bandwidth features of signals. In [23], Lu et al. enhanced their techniques through the usage of cyclic spectrum features. Nandi and Azzouz [24] proposed MLP neural networks with spectral feature sets for analogue, digital and combined modulation recognition. Their algorithms had inspired the foundation of a couple of commercial products or prototypes; examples of hardware implementation have been reported in a variety of applications, e.g. 4G software radio wireless networks [25], spectrum monitoring hardware [26, 27], etc.

Recently, several papers [28, 29, 30, 31] have proposed several novel methods in tackling the problem. Swami and Sadler [28] presented a comprehensive paper discussing their proposed higher-oder statistics (HOS) based hypothesis testing method. Wong and Nandi [30], extending the work of Azzouz and Nandi [17], proposed a combined set of spectral and statistical features in classifying 10 different modulation types. On the other hand, Mobasseri [29] suggested an alternative method by reconstruction the signal constellation shape using maximum likelihood estimation. The common trend among these algorithms seems to be recognising that the constellation shape is an unique signature among different modulation types.

The conventional classification methods exploit different instantaneous properties, e.g. instantaneous frequency, amplitudes, phase, etc. However, with the wide spread of usage of digital modulation nowadays, one could make use of the unique constellation diagram of each modulation type to discriminate among various digital modulations. Each constellation possesses different higher order cumulant values which can be estimated via its two dimensional histogram. An advantage of such a method is that it is immune to additive white and Gaussian noise as all higher-order cumulants for any Gaussian distributions are zero.

On the other hand, Mobasseri's [29] method provides an interesting alternative. By using clustering, his algorithm works under a non-supervised environment. Clustering also reduces the dimension of the input signal. To this end, it is parallel to feature extraction procedure in other pattern recognition algorithm. After clustering, Mobasseri proposed a Bayesian maximum likelihood (ML) method based on an image modeling method. However, the method suggested is sensitive to phase error, both static and non-static.

There are some other more generic forms of ML method that have also been reported. Wei and Mendel [32] showed that the constellation vector is a sufficient statistic for modulation recognition and that the ML classifier is optimal under ideal situations. Their work provides an upper bound for performance comparison for classifiers in non ideal situations. In another related work [33], they proposed a method based on fuzzy logic (FL) and had shown that, when fine tuned, FL based method corresponds to the ML solution. Meanwhile, Sills [34] extended the ML method to non-coherent environment which remains a challenging area of research in this field.

2.1.1 Applications of Modulation Classification

Conventionally, modulation classifier (MC) was devised for Communication Intelligence (COMINT) applications. Typical COMINT includes a receiver front-end, a modulation classification unit and a presentation stage. Types of receiver front-end include channelised receiver, scanning superheterodyne and an instantaneous frequency measurement (IFM). The front-end is important as most MCs require input from the decision front-end. Discussion of these is beyond the scope of this thesis. Similarly the presentation stage, e.g. demodulators, information extraction, deciphering, etc. need an accurate information from the MC to perform their operation and hence MC plays an important role in COMINT applications.

Examples of COMINT include the followings:

- Civilian Applications
  - Transmission control and monitoring
  - Signal confirmation
  - Interference identification
  - Spectrum management
- Military Application
  - Electronic support measures (ESM)

- Electronic counter measures (ECM)
- Threat detection and warning
- Target acquisition
- Homing

A more recently reported application of MC is in the area of software defined radio (SDR). Originally proposed for military purpose, recent proposals of SDR e.g. [35, 36] are gearing towards the public consumer market. Under the new software based architecture, SDR boasts high reconfigurability. Some SDRs have the ability of handling multiple modulation type as standard, enabled by the built-in MC unit. The work presented in this thesis favours a working environment that is like an SDR or adaptive modulation environment, in which, the application is channelised so information such as the carrier frequency and other channel parameters are mostly assumed to be known. However, the model introduced in later chapters will take into account the errors of such information, although these are thought to be minimal.

2.2 Preliminaries

The study of automatic classification of modulation signals is an interesting field as it requires some cross-disciplinary knowledge from different branches of engineering and computer science e.g. signal detection and identification, signal modulation, parameters estimation and pattern recognition. This thesis is a collection of work done in the above mentioned study, particularly of those using the statistical pattern recognition approach (includes both ANN and ML). Therefore in the following sections some background knowledge of digital communication as well as relevant mathematics are presented. Interested readers should refer to [37, 38, 17] for further in-depth discussions in communications and [39] for statistics related background.

2.2.1 Digital Communication Preliminaries

Communication Signals as Narrow Band Signals

Present day communication signals[1] are mostly narrow band signals, i.e. the signal bandwidth is relatively small in comparison with the center carrier frequency. For example, in radio communications, baseband signals are used to modulated carriers of higher frequencies spaced at intervals of the order of the signal bandwidth.

To enable the analysis of these signals, it is essential to convert these pass-band signals to baseband complex envelopes. Due to the typically high carrier frequency of such signals, a high sampling frequency rate is normally required. Along with this requirement, it is also desired to have high processing ability. Here, we shall look at ways to treat these signal efficiently while suppressing the largely irrelevant carrier oscillations.

Before proceeding, it needs to be emphasised that the task is to recognise various modulations but not to demodulate them. Therefore, details regarding demodulation are omitted.

Analytic Signal, Hilbert Transform and Complex Envelope

The Fourier Transform of any real signal is defined as:

$$\mathcal{F}(j\omega) = \int_{-\infty}^{\infty} f(t)exp(-j\omega t)dt \tag{2.1}$$

[1]with the exception of the recently launched 3G mobile communication system, which has a wide-band protocol, however in [40], it was shown that once the user code is decoded, the signal can be treated in the same way as narrowband signals.

For the purpose of clarity, we shall simply state $\mathcal{F}(j\omega)$ as $\mathcal{F}(\omega)$ for the rest of this thesis. It is trivial that, $\mathcal{F}(\omega)$ is conjugate symmetric, i.e. $\mathcal{F}(-\omega) = \mathcal{F}^*(\omega)$ where $*$ denotes complex conjugate.

Owing to this property, there is obvious redundancy of the negative frequencies as it is the mirror image of positive frequency. It is clear that we only need the one side of the signal spectrum. This representation of the signal is termed analytic signal.

An analytic signal $f_p(t)$ can be one that has the spectrum:

$$F_p(\omega) = F(\omega)[1 + \operatorname{sgn}(\omega)] \tag{2.2}$$

$$\operatorname{sgn}(\omega) = \begin{cases} 1 & \omega > 0, \\ 0 & \omega = 0, \\ -1 & \omega < 0. \end{cases} \tag{2.3}$$

In other words, the analytical is spectrum is twice the original spectrum for positive frequencies but zero for negative frequencies with DC component unchanged. Note that $f_p(t)$ is complex and it is sometimes referred to as the pre-envelope signal of $f(t)$. The real envelope of $f(t)$ can then be found via the magnitude of $f_p(t)$.

Rewriting Equation (2.2) as

$$F_p(\omega) = F(\omega) + j\hat{F}(\omega) \tag{2.4}$$

where

$$\hat{F}(\omega) = H(\omega)F(\omega) \tag{2.5}$$

with

$$H(\omega) = -j\text{sgn}(\omega) \tag{2.6}$$

A system with transfer function as in Equation (2.6) is called a Hilbert transformer, and a signal, $\hat{f}(t)$ with spectrum as in Equation (2.5), is the *Hilbert transform* of $f(t)$.

With this, the analytic signal $f_p(t)$ can be written as:

$$f_p(t) = f(t) + j\hat{f}(t) \tag{2.7}$$

In practice, $\hat{f}(t)$ can be realised by multiplying $f(t)$ with a quadrature filter, $\mathcal{F}_Q$, with impulse response, $r_Q(t)$ and complex gain $H(\omega)$. Thus we can write,

$$\begin{aligned}\hat{f}(t) &= \mathcal{F}_Q\{x(t)\} && (2.8)\\ &= x(t) * r_Q(t) && (2.9)\\ &= \int_{-\infty}^{\infty} x(t-\theta).r_Q(\theta)d\theta && (2.10)\end{aligned}$$

A complex envelope, $\tilde{f}(t)$, of a real signal, $f(t)$ is then defined as,

$$\begin{aligned}\tilde{f}(t) &= f_p(t)\exp(-j\omega_c t) && (2.11)\\ &= (f(t) + j\hat{f}(t))\ \exp(-j\omega_c t) && (2.12)\end{aligned}$$

where $\omega_c = 2\pi f_c$ in this context, refers to the carrier frequency in the passband.

Instantaneous Amplitude, Phase and Frequency

The instantaneous amplitude, $\tilde{a}(t)$ of real signal can be found using the magnitude of the analytic signal:

$$\tilde{a}(t) = \left(f^2(t) + \hat{f}^2(t)\right)^{\frac{1}{2}} \tag{2.13}$$

Similarly, the instantaneous phase, $\tilde{\phi}(t)$ can be calculated as follows:

$$\tilde{\phi}(t) = \begin{cases} \tan^{-1}(\frac{\hat{f}(t)}{f(t)}) & \text{if } f(t) > 0,\ \hat{f}(t) > 0, \\ \pi - \tan^{-1}(\frac{\hat{f}(t)}{f(t)}) & \text{if } f(t) < 0,\ \hat{f}(t) > 0, \\ \frac{\pi}{2} & \text{if } f(t) = 0, \hat{f}(t) > 0, \\ \pi + \tan^{-1}(\frac{\hat{f}(t)}{f(t)}) & \text{if } f(t) < 0,\ \hat{f}(t) < 0, \\ \frac{3\pi}{2} & \text{if } f(t) = 0, \hat{f}(t) < 0, \\ 2\pi - \tan^{-1}(\frac{\hat{f}(t)}{f(t)}) & \text{if } f(t) > 0,\ \hat{f}(t) < 0. \end{cases} \tag{2.14}$$

Finally, the instantaneous frequency, $\tilde{f}(t)$ is found by

$$\tilde{f}(t) = \frac{1}{2\pi}\frac{d\phi(t)}{dt} \tag{2.15}$$

Digital Modulations

There are general four major modulation schemes, namely amplitude shift keying (ASK), phase shift keying (PSK), frequency shift keying (FSK) and quadrature amplitude shift keying (QASK, also commonly known as quadrature amplitude modulation QAM). All modulation schemes are M-ary [2] derivation of the above four.

A digitally modulated signals in its complex envelope representation can be

[2]An expression used in digital communication, where M correspond to the number of symbols in the constellation diagram.

written as follows:

$$s(t) = A_m g_T(t) cos(2\pi f_m(t) + \phi_m(t)) \tag{2.16}$$

where A_m, f_m and ϕ_m are the message amplitude, message frequency and message phase respectively, in accordance with appropriate modulation techniques and g_T is the pulse shaping function. Consider the four main types of digital modulation techniques, i.e. Amplitude Shift Key (ASK), Phase Shift Key (PSK), Frequency Shift Key (FSK) and Quadrature Amplitude Modulation (QAM).

QAM, (e.g. QAM16, V29, V32, QAM64, etc.), is a newer type among the four and takes a slightly different form:

$$s(t) = s_I(t) g_T(t) cos 2\pi f_c t + s_Q(t) g_T(t) sin 2\pi f_c t \tag{2.17}$$

where $s_I(t)$ and $s_Q(t)$ are known as the in-phase and quadrature components of the signal respectively. Equation (2.17) is also known as the canonical representation for general digital modulation signals. Without loss of generality, we can take $g_T(t)$ as rectangular pulse function. Equation (2.17) can then be rewrote with its complex envelope representation:

$$s(t) = \big(s_I(t) + j s_Q(t)\big)\exp(-j2\pi f_c t) \tag{2.18}$$

2.2.2 Mathematical Preliminaries

Probability Theory

From a sample space, Ω, that contains a set of possible of outcomes where A being a particular set of those outcomes. In other words, A correspond with a subset of sample points in Ω. A probability measure is a function, $p(A)$, with A

as argument. It can be thought of the expected frequency of occurrence for A to be observed and possesses the following properties:

- $0 \leq p(A) \leq 1$
- $P(\Omega) = 1$
- If A and B are mutually exclusive events then

  $$P(A \cup B) = P(A) + P(B)$$

  or if they are not mutually exclusive

  $$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

Given a random variable x, the cumulative distribution function (CDF) is:

$$P_x(x) = \text{ probability that } X \leq x$$

and the derivative of the CDF is the probability density function (PDF) of x

$$p_x(x) = \frac{dP_X}{dx}$$

which has the following properties:

- $\int_{-\infty}^{\infty} p_x(x)dx = 1$
- $\int_a^b p_x(x)dx = P(a \leq X \leq b)$, and
- $p_x(x) \geq 0$

Higher Order Statistics

Some features and methods that we discuss in later chapters will invoke the use of higher order statistics (HOS). Gaussian density can be fully characterised using up to second order statistics (SOS). However, for other distributions, e.g. sub-Gaussian or super Gaussian, measures beyond SOS are needed. In digital communications, signals are non-Gaussian, and owing to the central limits theorem, the total effect of summation of different sources of noise can be model as a Gaussian distribution. This has motivated the use of HOS in many communication application since the mid-80's.

At the same time as interest in HOS was growing in signal processing community, ANN had become popular. An ANN is basically a large collection of distributed parallel non-linear processing units. Each processing unit, called a neuron, possess a chosen non-linearity. The non-linearity, for example a sigmoid function (or hyperbolic tangent function as it is also known), implicitly introduces HOS to the input data. For example, the sigmoid function, $\tanh(u)$ can be expanded into its Taylor series:

$$\tanh(u) = u - \frac{1}{3}u^3 + \frac{2}{15}u^5 - \ldots \tag{2.19}$$

For feed-forward ANN, e.g. the multilayer perceptrons (MLP), the scalar u is the inner product $u = \mathbf{w}^T\mathrm{x}$ where $\mathbf{x}$ denotes the input data, and $\mathbf{W}$ are the weights of the neurons. It is plain that HOS is implicitly utilised in this examples.

A successful example of HOS in recent signal processing would be independent components analysis (ICA), and its application in the problem of blind source separations (BSS). Other examples also includes blind channel equalisation, blind estimation of signal and noise ratio.

Expectation, Moments and Cumulants

Let x be a random variable of probability density function p_x, the expectation operator, $\mathrm{E}\{x\}$ for any function $g(x)$ of x is then defined as:

$$\mathrm{E}\{g(x)\} = \int_{-\infty}^{\infty} g(x)p_x(x)dx \qquad (2.20)$$

Moments are set of typical expectation used to characterise x. The j^{th} order moment α_j is written as:

$$\alpha_j = \mathrm{E}\{x^j\} = \int_{-\infty}^{\infty} x^j p_x(x)dx \qquad (2.21)$$

and the j^{th} order central moment μ_j of x is:

$$\mu_j = \mathrm{E}\{(x - m_x)^j\} = \int_{-\infty}^{\infty} (x - m_x)^j p_x(x)dx \qquad (2.22)$$

where m_x are the first order moment, better known as the mean of x. $\mu_0 = 1$ and $\mu_1 = 0$ are of little use to us, but μ_2 is the variance of x.

Skewness, the third order central moment μ_3, is a measure of asymmetry of the PDF of x. For a symmetrical PDF around the mean, skewness is equal to zero.

Kurtosis, on the other hand a measure of the length of the tail of the PDF, and it is zero for a Gaussian distribution. However, Kurtosis unlike skewness is not a central moment although it is related to the fourth order central moment μ_4. In fact, kurtosis is known as the the fourth order cumulant, κ_4 and it is defined as:

$$\kappa_4 = \mathrm{E}\{x^4\} - 3[\mathrm{E}\{x^2\}]^2 \qquad (2.23)$$

Here, we assumed x is zero mean.

Cumulants are related to moments by the natural logarithm of the moment generating function defined as below:

$$m.g.f(x) = \mathrm{E}\{\exp(j\omega x)\} = \int_{-\infty}^{\infty} \exp(j\omega x) p_x(x) dx \tag{2.24}$$

which incidentally is the Fourier Transform of the expectation operator.

For zero mean variable x, the first four cumulants are shown in Table 2.1.

Cumulants	Definition
κ_1	$\mu_1 = 0$
κ_2	$\mu_2 = \mathrm{E}\{x^2\}$
κ_3	$\mu_3 = \mathrm{E}\{x^3\}$
κ_4	$\mu_4 - 3\mu_2^2 = \mathrm{E}\{x^4\} - 3\mathrm{E}\{x^2\}^2$

Table 2.1: First four cumulants expressed in term of moments

From the table, it can be easily deduced that the first three cumulants are the same as the first three central moments, for the zero mean case. For the fourth order cumulant, kurtosis, there exists an normalised version which is more often used, $\hat{\kappa}_4$

$$\hat{\kappa}_4 = \frac{\mathrm{E}\{x^4\}}{\mathrm{E}\{x^2\}^2} - 3 \tag{2.25}$$

Sometimes the data is pre-whitened, thus the variance is unity, i.e. $\mathrm{E}\{x^2\} = 1$, in this case both version of kurtosis reduce to:

$$\hat{\kappa}_4 = \mathrm{E}\{x^4\} - 3 \tag{2.26}$$

which is the fourth order central moment with a bias.

Mean Vector, Correlation Matrix and Covariance Matrix

Let $f(\mathbf{x})$ denotes any function of a random vector of length N, $\mathbf{x} = [x_1, x_2, \ldots x_N]^T$, and $f(\mathbf{x})$ can either be a scalar, vector, or matrix. As with Equation 2.20, the *expectation* operator of $f(\mathbf{x})$ is defined as

$$\mathrm{E}\{f(\mathbf{x})\} = \int_{-\infty}^{\infty} f(\mathbf{x}) p_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \tag{2.27}$$

where $p_{\mathbf{x}}(\mathbf{x})$ again denotes the probability density function (p.d.f.) of $\mathbf{x}$. The integral in carried out on each component of $\mathbf{x}$, $x_1, x_2, \ldots, x_n$ yielding a new quantity of the same dimension.

The first order moment, is called *mean vector*, $\mathbf{m_x}$ of $\mathbf{x}$, is defined as the expectation of $\mathbf{x}$:

$$\mathbf{m_x} = \mathrm{E}\{\mathbf{x}\} = \int_{-\infty}^{\infty} \mathbf{x} p_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} \tag{2.28}$$

Correlation matrix is defined by the second order expectation of $\mathbf{x}$:

$$\mathbf{R_{xx}} = \mathrm{E}\{\mathbf{x}\mathbf{x}^T\} \tag{2.29}$$

A correlation matrix is a positive semi-definite matrix, i.e. $\mathbf{a}^T \mathbf{R_{xx}} \mathbf{a}$ is greater than zero for all nonzero vector $\mathbf{a}$, and in practice it is usually positive definite, provided the sample size of $\mathbf{x}$ is large enough.

We define central moments for random vectors as we did for the random variables, however, it is plain that first order central moment for random vectors are irrelevant. Second order central moment for $\mathbf{x}$ is termed as the covariance matrix, and the covariance operation is sometimes denoted by, $\mathrm{cov}(\mathbf{x})$. Covariance matrix $\mathbf{C_{xx}}$ is the expectation of the central moment of $\mathbf{xx}$

$$\mathbf{C_{xx}} = \mathrm{E}\{(\mathbf{x} - \mathbf{m_x})(\mathbf{x} - \mathbf{m_x})^T\} \tag{2.30}$$

Cross covariance for two vectors, $\text{cov}(\mathbf{x}, \mathbf{y})$ can be similarly defined:

$$\mathbf{C}_{\mathbf{xy}} = \mathrm{E}\{(\mathbf{x} - \mathbf{m}_{\mathbf{x}})(\mathbf{y} - \mathbf{m}_{\mathbf{y}})^T\} \tag{2.31}$$

Each component $\mathbf{C}_{\mathbf{xy}}^{i,j}$ in $\mathbf{C}_{\mathbf{xy}}$ then represents the cross-covariance between $\mathbf{x}^i$ and $\mathbf{y}^j$.

$$\mathbf{C}_{\mathbf{xy}} = \mathrm{E}\{(\mathbf{x} - \mathbf{m}_{\mathbf{x}})(\mathbf{y} - \mathbf{m}_{\mathbf{y}})^T\} \tag{2.32}$$

Likewise, the cross-correlation matrix is defined as:

$$\mathbf{R}_{\mathbf{xy}} = \mathrm{E}\{\mathbf{x}\mathbf{y}^T\} \tag{2.33}$$

Chapter 3

Feature Extraction and Data Preprocessing

3.1 Data Model for Digital Modulation

To build up a working model for the problem, the working environment is assumed to be coherent and synchronous. This implies that the carrier frequency, data baud rate and channel coefficient have been adequately established. In practice, these parameters are sometimes not known, for example in a furtive environment, where the signal is intercepted stealthily and these parameters have to be estimated. A high sampling rate[1] is also usually required for adequate recovery of these paratmeter. Nevertheless, in other civil applications, these parameters are sometimes available. An example of this is the combined orthogonal frequency division multiplexing (OFDM) - code division multiple access (CDMA) technique. In OFDM-CDMA using adaptive modulation [41, 42] where the modulation type of the signal is depending on other operating factors, it can be assumed that the

[1]The features proposed in this work are extracted using symbol based calculation, therefore changes of sampling rate should not affect the training of ANN. Nevertheless, a high sampling rate and fast processing power is always desired.

carrier frequency and sample baud rate are known *a priori.*

In the case where the carrier frequency is unknown, a coarse estimate of the carrier frequency can be obtained using the power spectral density (PSD) of the intercepted signal. The baseband in-phase (I) and quadrature (Q) part of the signal can then be recovered through the quadrature filter (i.e. Hilbert transformer in Chapter 2). The carrier frequency can be refined via various high resolution spectrum analysis methods [43, 44]. The symbol baud rate can be estimated through a tracking loop, and symbol timing can be obtained via standard fractional sampling schemes [37, Chapter 6].

Here, we also assumed that the channel effects on the received signal have been equalised. A constellation-independent algorithm has to be used in this task as by definition the underlying constellation type is unknown. Examples of these blind equalisation algorithms include the Godard's algorithm [45], and the constant modulus algorithm (CMA), [46, 47].

Assuming that the above mentioned tasks have been carried out, we introduce the following model for the received data:

$$y(n) = A \cdot \sum_{\ell=-\infty}^{\infty} x(\ell)h(nT - \ell T + \epsilon_T T) \cdot \exp(j2\pi f_o T_n + j\theta_n) + g(n) \tag{3.1}$$

where $x(\ell)$ is the signal sequence, A is an unknown amplitude factor, $h(\cdot)$ represents the residual baseband channel effects, T is the symbol spacing, ϵ_T is the timing error, f_o is the constant frequency offset for the received sequence and θ_n is the phase jitter which vary from sample to sample in the recovered sequenced. $g(n)$ is the complex additive white and Gaussian noise (AWGN).

Here, we further assumed that the channel effects and symbol timing errors are negligible, and the phase jitters are sufficiently small, we can then take a simpler model as follows:

$$y(n) = Ax(n)\exp(-j2\pi f_o) + g(n) \tag{3.2}$$

Although AWGN is assumed in this model, $g(n)$ can equally be from a coloured process too. $x(n)$ is the complex envelope of the received samples which takes the form of (2.11). The real and imaginary part of $x(n)$ correspond to the I and Q components of the signal. By plotting the I and Q components of the signal with a two dimensional scatter plot, we obtained the estimated constellation diagram for the received signal. The constellation diagram is unique to a particular modulation type (see Figure 3.1).

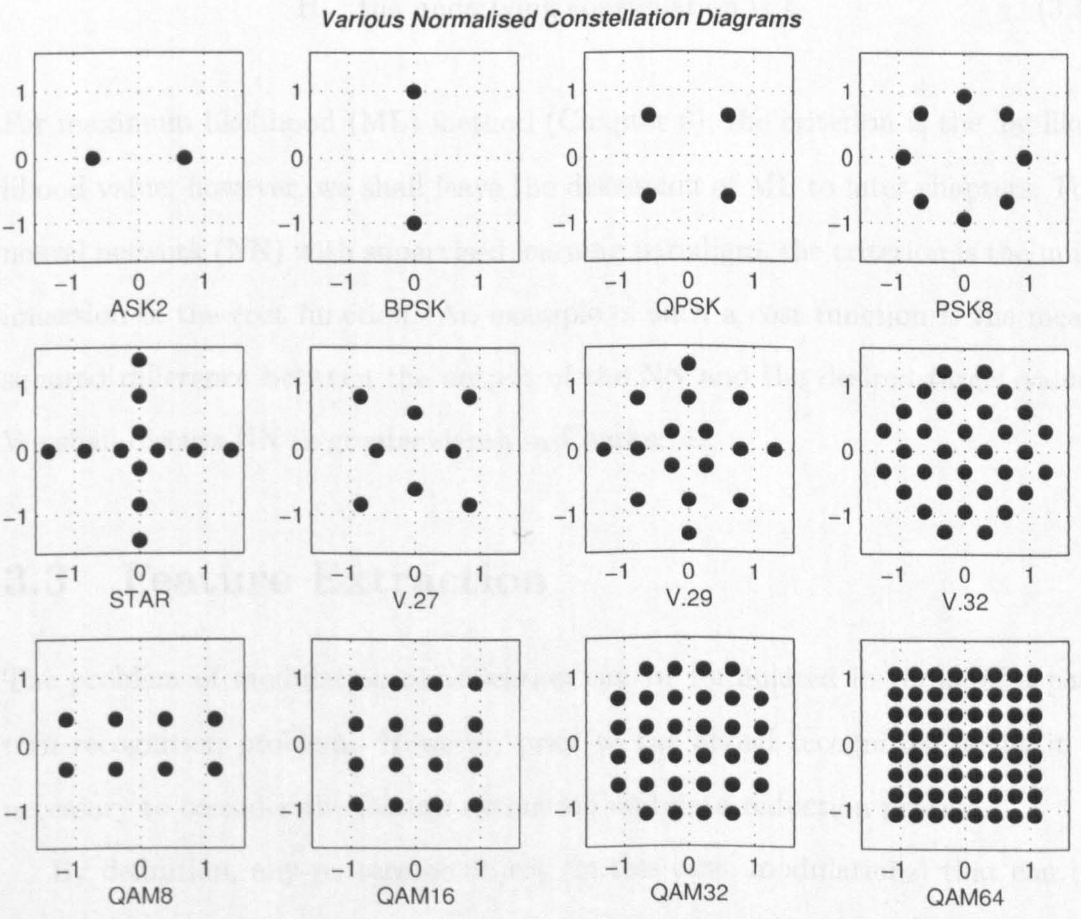


Figure 3.1: The 12 digital modulation types considered in this work

3.2 Problem Statement

Given N samples of received signal, $y(n)$, and a set of c possible modulation types, I_j, as follows:

$$\mathrm{I}_j = \{x_{j1}, x_{j2}, \dots, x_{jM_j}\}, \qquad j = 1, 2, \dots, c \tag{3.3}$$

where x_{jk} is a point in the constellation j and M_j is the total number of points that constellation; by normalising both $y(n)$ and I_j to unit power, we can then take A to be unity.

Given a set of K received signals, denoted by vector $Y_k = \{y_1, y_2, \dots, y_K\}$, the task is to choose the hypothesis, H_j, based on some pre-defined criterion:

$$\mathrm{H}_j : \text{the underlying constellation is } \mathrm{I}_j \tag{3.4}$$

For maximum likelihood (ML) method (Chapter 6), the criterion is the log likelihood value, however, we shall leave the discussion of ML to later chapters. For neural network (NN) with supervised learning paradigm, the criterion is the minimisation of the cost function. An example of such a cost function is the mean squared difference between the output of the NN and the desired target value. We shall discuss NN in greater depth in Chapter 4.

3.3 Feature Extraction

The problem of modulation classification can be formulated in terms of a pattern recognition problem. However, prior to the actual recognition stage, it is necessary to consider the feature extraction and data reduction process.

By definition, any pattern or object (in this case, modulations) that can be classified possesses some discriminative properties that differentiate between itself and other patterns or objects. These properties are called features (analogous to the human face features) in pattern recognition problem. Three type of features exists in general, i.e. physical features, structural features and mathematical features.

Examples of physical features are colour of an object, its fragrance, it materials, etc. Structural features includes structural properties of the objects such as its shape, height, weight, textures, volume, etc. The third category (the mathematical features), which might be the most relevant in this context, are derived properties of the object. E.g. the statistical mean, statistical variance, Fourier Series, eigenvalues or eigenvectors of covariance matrix etc. A classifier is essentially a mathematical algorithm dealing with any computable features. An often desirable characteristic for a given feature is invariance. Any good feature should be invariant toward scaling, dilation, transformation, etc. Therefore, care need to be exercised when selecting and defining features.

Various features for modulation classification have been proposed as mentioned previously (Chapter 2). Among which, Azzouz and Nandi [48] proposed a spectral feature set for digital communication signal. Azzouz and Nandi tackled various modulation types, namely ASK2/4, PSK2/4, and FSK2/4. However, while the spectral feature set recognised the fact that information is hidden in either the amplitude, phase and frequency spectrum of the signal, and were useful for their application, the newer modulation schemes like M-ary QAM signals contain both information in amplitude and phase spectrum. A HOS based hierarchical algorithm has been proposed by Swami and Sadler [28], and has shown robustness and proven performance in multiple scenarios.

Some of the features that Swami and Sadler proposed are complex features, although they appeared to be real valued if the frequency offset is small. Therefore, complex valued NN are needed if these features is to be used instead of the standard NN. Later in this chapter, the author proposes a set of alternative HOS based real valued features that can be applied directly to a standard NN. The examples given in Swami and Sadler's literature addresses particular set of modulation types, however, most of their examples contains a small number of modulation types (c = 2,4) and owing to the hierarchical hypothesis testing method, a particular set of modulation types require a different formation of the classifier. Great care and experience are exercised for the each formation of the solution. When a new modulation type is added to the system, the user needs to re-examine the feature threshold manually which can be equally cumbersome and time consuming. However, by using a NN solution, the user simply needs to retrain the network by adding the new training data and target class label to the existing database.

With any feature extraction procedure, the number of features used is normally kept to a minimum, as the input dimension of the pattern classifier is determined by the number of features. The higher the input dimension, the more the number of training samples that is required. This is known as *the curse of dimensionality.* It is the analogous to the estimation problem in statistics, in that one needs sufficient samples to generate a meaningful statistical estimate. Feature selection algorithms (Chapter 5) can be used to choose the feature subset that gives the best performance.

3.4 Data Pre-processing

Prior to feature extraction, it is customary to take certain measures to ensure the integrity of the data. Such measures are commonly known to as data pre-processing. Some examples of these pre-processing techniques are outlier removal, data normalisation and missing data treament.

Outliers are points that lie far away from the statistical mean of a random variable. The presence of outliers also leads to large errors during training, which can be costly to the classification process. Normally, a threshold is determined (e.g. a multiple of standard deviation) to remove possible outlier.

Apart from the problem of outliers, it is also possible that the amount of available data is not the same for all features. This is the case of missing data. This can be a significant problem when the data set is small (a problem typically found in genomic signal processing). Otherwise, the examples with missing data can be disregarded. In practice, the missing data is sometimes predicted through heuristic methods (see [49, 50]).

Among these pre-processing method, data normalisation is the most relevant, as it ensures invariance of certain features. One common normalisation method is that the signal/data is ensured to be of zero mean and of unit variance. Mathematically, this is achieved through the following:

$$\hat{x}_k(i) = \frac{x_k(i) - \bar{x}_k}{\sigma_{x_k}} \tag{3.5}$$

where i is the discrete time index, $\bar{x}_k$ and σ_{x_k} are the mean and standard deviation of k^{th} signal examples respectively.

3.5 Cumulants

In Section 2.2.2, we introduced some higher order statistics for the case of a random variable, where the definition of the moments and cumulants of a random variable were given. In a similar fashion, higher order cross-cumulants for random variables can be defined as follows:

Given N samples of a random variable, $x_i(n)$, arranged in a vector format,

$\{\mathbf{x}_i = [x_i(1), x_i(2), \ldots, x_i(N)]^T\}$, and $E\{\cdot\}$ denote the statistical expectation. The second, third and fourth order cross cumulants[2] can be defined (c.f. Chapter 3 in [51]) as follows:

$$\begin{aligned}\mathrm{Cum}_{\mathbf{x}_1,\mathbf{x}_2} &= E\{\mathbf{x}_1,\mathbf{x}_2\} \\ &= \frac{1}{N}\sum_{n=1}^{N} x_1(n)x_2(n)\end{aligned} \tag{3.6}$$

$$\begin{aligned}\mathrm{Cum}_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3} &= E\{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3\} \\ &= \frac{1}{N}\sum_{n=1}^{N} x_1(n)x_2(n)x_3(n)\end{aligned} \tag{3.7}$$

$$\begin{aligned}\mathrm{Cum}_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3,\mathbf{x}_4} &= E\{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3,\mathbf{x}_4\} \\ &\quad -E\{\mathbf{x}_1,\mathbf{x}_2\}E\{\mathbf{x}_3,\mathbf{x}_4\} \\ &\quad -E\{\mathbf{x}_1,\mathbf{x}_3\}E\{\mathbf{x}_2,\mathbf{x}_4\} \\ &\quad -E\{\mathbf{x}_1,\mathbf{x}_4\}E\{\mathbf{x}_2,\mathbf{x}_4\} \\ &= \frac{1}{N}\sum_{n=1}^{N} x_1(n)x_2(n)x_3(n)x_4(n) \\ &\quad -\mathrm{Cum}_{\mathbf{x}_1,\mathbf{x}_2}\mathrm{Cum}_{\mathbf{x}_3,\mathbf{x}_4} \\ &\quad -\mathrm{Cum}_{\mathbf{x}_1,\mathbf{x}_3}\mathrm{Cum}_{\mathbf{x}_2,\mathbf{x}_4} \\ &\quad -\mathrm{Cum}_{\mathbf{x}_1,\mathbf{x}_4}\mathrm{Cum}_{\mathbf{x}_2,\mathbf{x}_3}\end{aligned} \tag{3.8}$$

Under ideal environment, the signal model introduced in Equation (3.2) is only affected by additive Gaussian noise, $g(n)$, and the phase offset. Assuming that the

[2]These cumulants are derived using central moments, and since the signal is of zero mean (first order moment), hence for clarity, all relevant products of first order moment in Equation (3.7) had been dropped.

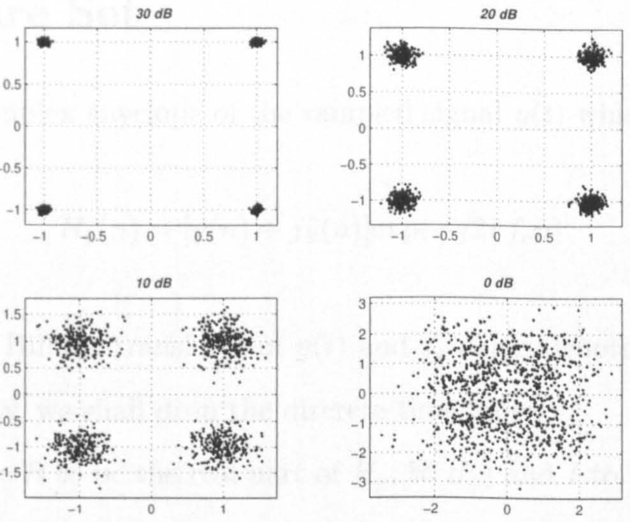


Figure 3.2: Corruption of the signal's constellation under AWGN at different SNR

operating environment is coherent, the received signal is then only corrupted by $g(n)$. Figure 3.2 shows how the noise affected the reconstructed I-Q diagram. For Gaussian noise, higher order cumulants ($n > 2$) are equal to zero; therefore the estimated higher order cumulants of the received signal, $y(n)$, can be attributed to the higher order cumulants of the source signal, $x(n)$, as illustrated below using the fourth order cumulants:

$$\begin{aligned} \mathrm{Cum}_{\mathrm{y_1,y_2,y_3,y_4}} &= \mathrm{Cum}_{\mathrm{x_1,x_2,x_3,x_4}} + \mathrm{Cum}_{\mathrm{g_1,g_2,g_3,g_4}} \\ &= \mathrm{Cum}_{\mathrm{x_1,x_2,x_3,x_4}} + 0 \end{aligned} \tag{3.9}$$

This provides an interesting prospect for using higher order cumulants as features to classify different modulation types.

3.6 Feature Set

Let H_y be the complex envelope of the sampled signal $y(t)$ which is defined by:

$$H_y(n) = [y(n) + j\hat{y}(n)]\exp(-j2\pi f_c n) \tag{3.10}$$

where $\hat{y}(t)$ is the Hilbert transform of $y(t)$ and f_c again denotes the carrier frequency. For clarity, we shall drop the discrete time index.

We then define R to be the real part of H_y, $\Re(H_y)$ and I to be the imaginary part of H_y, $\Im(H_y)$. For elegance, we adopt the following convention for n^{th} order cumulants:

$$\kappa_{n-p,p} = \mathrm{Cum}_{\underbrace{R\ldots R}_{n-p}\underbrace{I\ldots I}_{p}} \tag{3.11}$$

Thus, the following features are introduced:

$$\{\kappa_{20}, \kappa_{11}, \kappa_{02}, \kappa_{30}, \kappa_{21}, \kappa_{12}, \kappa_{03}, \kappa_{40}, \kappa_{31}, \kappa_{22}, \kappa_{13}, \kappa_{04}\} \tag{3.12}$$

which are the second, third and fourth order cumulants and cross cumulants of the real and imaginary parts of the signal.

As most of the constellation types considered (see Figure 3.1) here have four-fold symmetry (with the exception of ASK2, BPSK and QAM8), it follows that theoretically, the features listed in Equation 3.13 for these summetrical modulation types are null features, i.e. their value should be zero, and hence non-discriminating.

$$\{\kappa_{11}, \kappa_{30}, \kappa_{21}, \kappa_{12}, \kappa_{03}, \kappa_{31}, \kappa_{13}\} = \{0\} \tag{3.13}$$

Table 3.1 shows the value of all the values of the elements in Equation 3.12, calculated from the ideal noise free constellation diagrams (SNR $= \infty$). The result is in accordance with Equation 3.13. It is normally assumed the received signal is normalised to unit power. In practice self-normalised cumulants are often used to avoid the scaling problem. Owing to the four-fold symmetry properties, κ_{20} and κ_{02} are the same (again, with the exception of the three constellations mentioned earlier). Therefore, the fourth order cumulants are normalised by their second order cumulants. To generalise for modulation types which do not process four-fold symmetry, the higher value between κ_{20} and κ_{02} is taken to be the normalising cumulant, κ_{norm}, i.e.

$$\kappa_{norm} = \max\{\kappa_{20}, \kappa_{02}\} \tag{3.14}$$

and

$$\tilde{\kappa}_{n-p,p} = \left[\frac{\kappa_{n-p,p}}{\kappa_{norm}}\right] \tag{3.15}$$

After normalisation, the second order cumulants are not kept since they are now non-discriminating. Therefore, the final feature set comprises the following fourth order cumulants:

$$\{\tilde{\kappa}_{40}, \tilde{\kappa}_{22}, \tilde{\kappa}_{04}\} \tag{3.16}$$

In Table 3.2, we consider the theoretical values for the following phase and quadrature modulation types:

· ASK2	· BPSK	· QPSK	· PSK8
· STAR	· V.27	· V.29	· V.32
· QAM8	· QAM16	· QAM32	· QAM64

These values were again calculated using the ideal constellation diagrams of each individual modulation. Besides, the standard deviations of these values at 0, 10

Cumulants	ASK2	BPSK	QPSK	PSK8	STAR	V.27	V.29	V.32	QAM8	QAM16	QAM32	QAM64
κ_{20}	0.50	0.00	0.38	0.44	0.46	0.44	0.47	0.48	0.73	0.47	0.48	0.49
κ_{11}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
κ_{02}	0.00	0.50	0.38	0.44	0.46	0.44	0.47	0.48	0.15	0.47	0.48	0.49
κ_{30}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
κ_{21}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
κ_{12}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
κ_{03}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
κ_{40}	−0.50	0.00	−0.28	−0.29	0.10	−0.30	−0.14	−0.22	−0.72	−0.30	−0.27	−0.30
κ_{31}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
κ_{22}	0.00	0.00	0.00	−0.10	−0.21	0.05	−0.12	−0.10	0.00	0.00	−0.06	0.00
κ_{13}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
κ_{04}	0.00	−0.50	−0.28	−0.29	0.10	−0.30	−0.14	−0.22	−0.04	−0.30	−0.27	−0.30

Table 3.1: Theoretical cumulant values for various modulation types

and 20 dB signal-to-noise ratio (SNR) are also presented here (Tables 3.5, 3.4, 3.3). The SNR is defined mathematically as follows:

$$\mathrm{SNR} = 10\log\left[\sum_{n=1}^{N}\frac{y^2(n)}{g^2(n)}\right] \tag{3.17}$$

where $y(n)$ and $g(n)$ are the signal and the Gaussian noise respectively.

3.7 Conclusion

This chapter presented a practical mathematical model of a received signal with unknown modulation via an AWGN channel. This complex model which takes into account various non-idealities upon interception of signals can be simplifed under the assumption of a coherent channel. The problem definition for modulation classification was also stated in the first half of this chapter.

Feature extraction is an integral part of the pattern recognition problem. An important criterion for an ideal feature is invariance against translation, transformation, noise corruption, rotation etc. Such features are difficult to obtain in real life. However, if the underlying mechanism of a problem can be clearly understood, it is then easier to find features that are near to the ideal conditions.

In this chapter, a small feature set based on HOS is introduced. Communication signals are often corrupted by Gaussian noise. In general, HOS provides a good degree of immunity against additive Gaussian noise, as the higher order cumulants (of order > 2) are zeros. Although, in practice, owing to the finite samples scenario, variability can still be found in the estimates of these HOS.

HOS provides good descriptive measures for 2D constellation diagrams. It provides an intuitive method for discriminating among different modulation type with different constellation diagrams. A statiscal feature set, based on the fourth

Features	ASK2	BPSK	QPSK	PSK8	STAR	V.27	V.29	V.32	QAM8	QAM16	QAM32	QAM64
$\tilde{\kappa}_{40}$	−1.00	0.00	−0.75	−0.66	0.21	−0.68	−0.29	−0.46	−0.99	−0.64	−0.55	−0.61
$\tilde{\kappa}_{22}$	0.00	0.00	0.00	−0.22	−0.46	0.12	−0.26	−0.21	0.00	0.00	−0.12	0.00
$\tilde{\kappa}_{04}$	0.00	−1.00	−0.75	−0.66	0.21	−0.68	−0.29	−0.46	−0.06	−0.64	−0.55	−0.61

Table 3.2: Theoretical values for the final feature set calculated with ideal noise free constellation diagrams

Standard Deviation	**ASK2**	**BPSK**	**QPSK**	**PSK8**	**STAR**	**V.27**	**V.29**	**V.32**	**QAM8**	**QAM16**	**QAM32**	**QAM64**
$std(\tilde{\kappa}_{40})$	0.02	0.00	0.04	0.06	0.07	0.04	0.06	0.06	0.04	0.05	0.06	0.05
$std(\tilde{\kappa}_{22})$	0.00	0.00	0.00	0.01	0.02	0.01	0.02	0.01	0.00	0.02	0.01	0.02
$std(\tilde{\kappa}_{04})$	0.00	0.02	0.04	0.06	0.07	0.04	0.06	0.06	0.01	0.05	0.06	0.05

Table 3.3: Standard deviation of proposed features at 20 dB SNR

Standard Deviation	**ASK2**	**BPSK**	**QPSK**	**PSK8**	**STAR**	**V.27**	**V.29**	**V.32**	**QAM8**	**QAM16**	**QAM32**	**QAM64**
$std(\tilde{\kappa}_{40})$	0.07	0.02	0.06	0.08	0.07	0.06	0.06	0.06	0.05	0.05	0.06	0.05
$std(\tilde{\kappa}_{22})$	0.02	0.02	0.02	0.03	0.02	0.03	0.02	0.02	0.01	0.02	0.02	0.02
$std(\tilde{\kappa}_{04})$	0.02	0.07	0.06	0.07	0.07	0.06	0.06	0.06	0.01	0.05	0.06	0.05

Table 3.4: Standard deviation of proposed features at 10 dB SNR

Standard Deviation	**ASK2**	**BPSK**	**QPSK**	**PSK8**	**STAR**	**V.27**	**V.29**	**V.32**	**QAM8**	**QAM16**	**QAM32**	**QAM64**
$std(\tilde{\kappa}_{40})$	0.11	0.08	0.10	0.10	0.10	0.09	0.08	0.08	0.10	0.08	0.07	0.06
$std(\tilde{\kappa}_{22})$	0.04	0.04	0.04	0.04	0.03	0.04	0.03	0.03	0.03	0.04	0.03	0.03
$std(\tilde{\kappa}_{04})$	0.08	0.12	0.10	0.10	0.09	0.10	0.09	0.08	0.05	0.08	0.07	0.06

Table 3.5: Standard deviation of proposed features at 0 dB SNR

order cumulants of the I-Q diagram, was devised. The theoretical values shown in Table 3.2 suggest that discimination is possible using the proposed statistical feature set. The tightly bounded standard deviations of these features in high SNR (> 10 dB) give an optimistic view for the actual classification process. Although when the noise gets higher, the bound of the standard deviaton also lossen. This is understandable judging from Figure 3.2.

Chapter 4

Digital Modulation Classification using Neural Networks

Artificial neural networks (ANN) first overwhelmed the signal processing society in the mid-80s. Today the ANN is widely used in many digital communication application [52]. It applications ranges from simple channel estimation to MIMO channel equalisation. Other examples can also be found in control engineering, condition monitoring, image processing, etc. ANN can be grouped into two major categories by their learning paradigms, i.e. supervised learning (learning with a teacher) or unsupervised learning (clustering). Unsupervised learning is widely used in applications such as data representation, vector quantisation, data mining, etc; while supervised learning is commonly used in non-linear regression and pattern classification. In this work, the application of supervised learning based neural networks in digital modulation classification is investigated.

Supervised neural networks are popular choices in recent developments of modulation classification. It is a preferred alternative approach to decision theoretic (DT) based algorithms [17] when training examples are available. ANN based algorithms have the ability to learn the optimal threshold via training. This differs from some algorithms based on the DT approach where many thresholds of the key feature values needed to be chosen carefully with detailed analyses. Another major advantage of supervised ANN over DT is that the key features can be applied all at the same instance. For DT however, the order of evaluating certain key features are crucial, as it can give rise to differing performance. Training the ANN requires all features must be available prior to the classification phase. This may be time-consuming if the number of key features is large and is extracted serially. Nevertheless, this issue can be overcame through parallel processing of key features as often they do not rely on the presence of other key features.

When an ANN classifier is trained, i.e. the underlying structure and parameters are fixed adaptively through the training data, it may be used for on-line analysis. This is owing to the computational simplicity of the ANN. Besides, when changes occurred to the operating environment (e.g. changes of SNR), ANN can be easily retrained for the new operating condition. Most ANNs also demonstrate robustness and tolerances against faults.

In general, an ANN classifier can be viewed as shown in Figure 4.1. A set of pre-defined features are calculated during the feature extraction stage. The features are then fed through the pre-trained neural networks which consists an interconnection of weighted non-linear processing units called neurons. These weighted neurons form a set of non-linear discriminant functions which will output a set of soft decisions. A final classification stage transforms these soft decisions into hard decisions where the output is associated to one modulation type in the context of autmatic digital modulation classification (ADMC).

The operation of each ANN, in general, consists of a training phase and a testing phase. Depending on the training methodology adopted, a validation phase is sometimes required. In the training phase, the weight of the neurons are

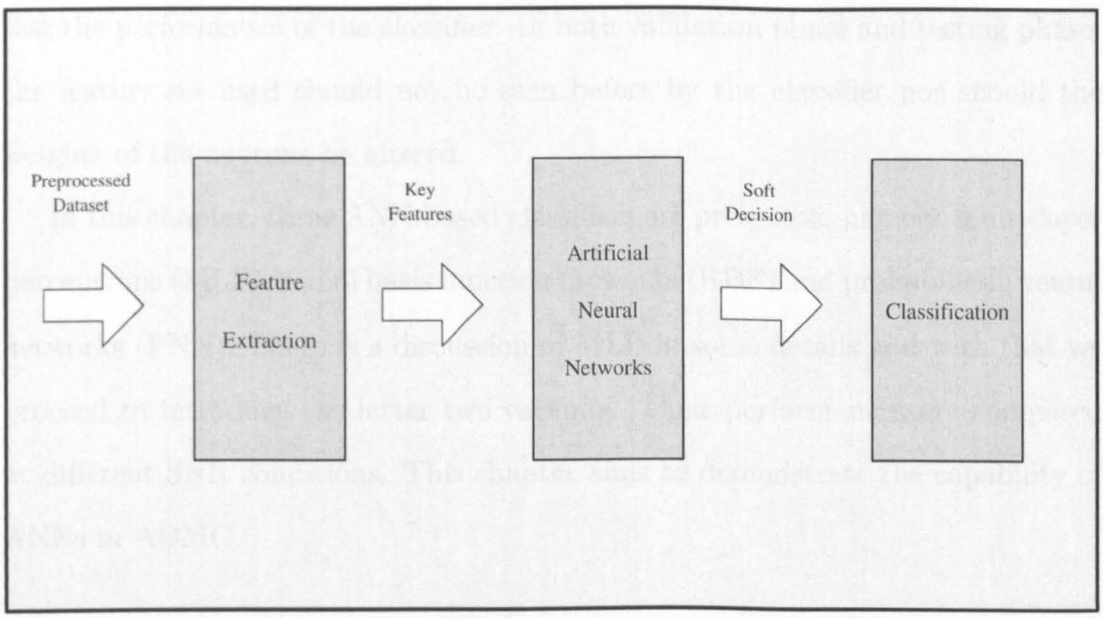


Figure 4.1: ANN based modulation classifier

constantly adjusted with the aid of error information fed back through a feedback path.

In the validation phase, if present, a separate set of validation data is used to check the generality of the classifier at the end of each training epoch (iteration). A classifier with good generalisation is one that has good classification for all samples, including those it has never seen before. It is essential that a classifier performs well for all three datasets (training, validation and testing). During validation phase, if the classifier's error index of the validation set increases consistently for a pre-defined period, despite the error index of its training set still decreassing, the classifier is said to be "overfitting" the training set. Hence, the training process is interuptted. Subsequently, the weights of the classifier are reverted to the previous weight values that gave the smallest validation error index. This method is called "generalisation via early stopping" and is commonly used in back-propagation learning based neural networks.

Once the training phase is completed, a new feature set is then supplied to test the performance of the classifier. In both validation phase and testing phase, the feature set used should not be seen before by the classifier nor should the weights of the neurons be altered.

In this chapter, three ANN based classifiers are presented, namely, multi-layer perceptrons (MLP), radial basis function networks (RBF) and probabilistic neural networks (PNN). There is a discussion of MLP in some details and with that we proceed to introduce the latter two variants. Their performances are compared at different SNR conditions. This chapter aims to demonstrate the capability of ANNs in ADMC.

4.1 Multi-layer Perceptrons Classifier

MLP (see Figure 4.2) is a feed-forward structure of interconnection of individual non-linear parallel computing units called neurons. Inputs are propagated through the network layer by layer and MLP gives a non-linear mapping of the inputs at the output layers.

We can write MLP mathematically as:

$$y_k(n) = \phi_2\Big(\sum_{j=1}^{q} w_{kj}\phi_1\Big(\sum_{i=0}^{p} w_{ji}x_i(n)\Big)\Big) \tag{4.1}$$

where n is the sample number, subscript k denotes the output nodes, subscripts i and j denote hidden nodes and inputs nodes respectively; p and q are the total number of neurons available in layer i and j respectively. Note that the activation functions, $\phi_{1,2}$, can vary for different layers of neurons.

Classification generally consists of two phases - training and testing. A paired training input and target output are presented at each training epoch, output errors and weights are calculated according to the chosen learning algorithm.

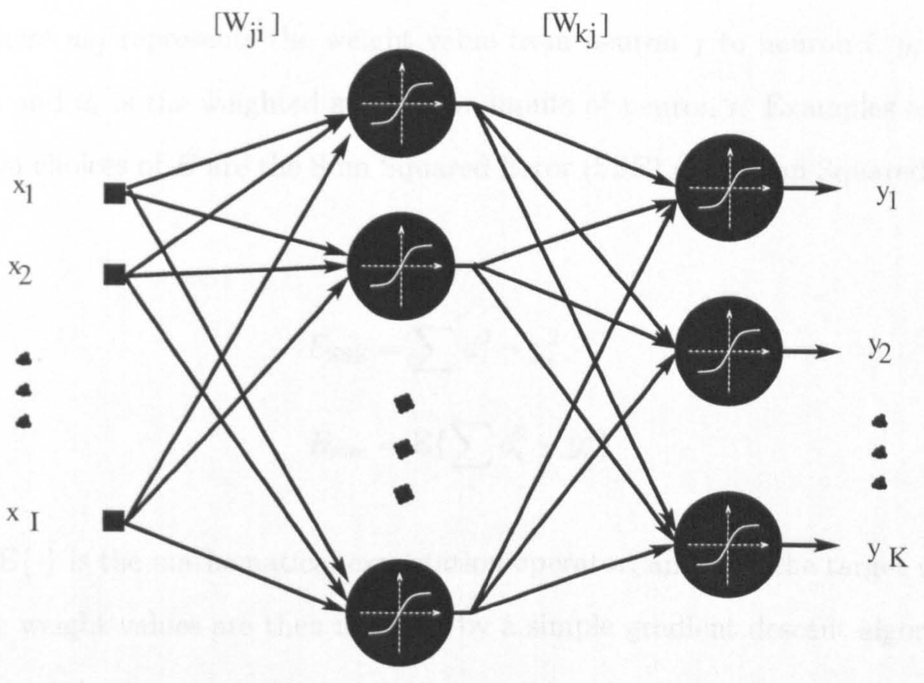


Figure 4.2: A two-layer MLP with sigmoid activation

For a batch training algorithm, weights are updated once every training epoch, meaning a full run of training sample, while in adaptive training, weights are updated every training sample. Learning algorithms shall be discussed in more depth in the next section.

4.1.1 Learning Algorithms for MLP

The MLP is a relatively mature branch of ANN and there are a number of efficient training algorithms. Azzouz and Nandi adopted the standard back-propagation algorithm (BP) [53]. They also used BP with momentum and adaptive learning rate to speed up the training time required.

BP algorithms implement generalised chained rules repetitively to calculate the changes of each weight with respect to the error function, E.

$$\frac{\delta E}{\delta w_{ij}} = \frac{\delta E}{\delta y_i}\frac{\delta y_i}{\delta u_i}\frac{\delta u_i}{\delta w_{ij}} \tag{4.2}$$

and where w_{ij} represents the weight value from neuron j to neuron i, y_i is the output and u_i is the weighted sum of the inputs of neuron i. Examples of some common choices of E are the Sum Squared Error (SSE) and Mean Squared Error (MSE):

$$E_{\text{SSE}} = \sum_i d_i^2 - y_i^2 \tag{4.3}$$

$$E_{\text{mse}} = \mathrm{E}\{\sum_i d_i^2 - y_i^2\} \tag{4.4}$$

where $\mathrm{E}\{\cdot\}$ is the mathematical expectation operator, and d_i is the target vector.

The weight values are then updated by a simple gradient descent algorithm:

$$w_{ij}(t+1) = w_{ij}(t) - \epsilon \frac{\delta E}{\delta w_{ij}}(t) \tag{4.5}$$

The learning rate parameter, ϵ, is analogous to the step size for a least mean square (LMS) adaptive filter, where a higher learning rate means a faster convergence, but with risk of oscillation. On the other hand, a value too small will take too much time to achieve convergence. An adaptive learning rate variant of BP takes account of this problem by updating the learning rate adaptively. By doing so, one also avoids trapping in a local minimum.

A BP algorithm with momentum adds an extra momentum parameter, μ, to the weight changes:

$$\Delta w_{ij}(t+1) = -\epsilon \frac{\delta E}{\delta w_{ij}}(t) + \mu \frac{\delta E}{\delta w_{ij}}(t-1) \tag{4.6}$$

This takes account of the previous weight changes and leads to a more stable algorithm and accelerates convergence in shallow areas of the cost function.

In recent years, new algorithms have been proposed for network training.

However, some algorithms require much computing power to achieve good training, especially when dealing with a large training set. Although these algorithms require very small numbers of training epochs, the actual training time for a epoch is much longer compared with BP algorithms. An example would be the Levenberg-Marquardt Algorithm (LM) [54].

Here, we consider the BP algorithms and the resilient back-propagation algorithm (RPROP) proposed by Riedmiller and Braun [55] in 1993. Basically, unlike BPs, RPROP only considers the sign of derivatives as the indication for the direction of the weight update. In doing so, the size of the partial derivative does not influence the weight step.

The following equation shows the adaptation of the update values of Δ_{ij} for the RPROP algorithm. For initialisation, all Δ_{ij} are set to small positive values.

$$\Delta_{ij}(t) = \begin{cases} \eta^{+} * \Delta_{ij}(t-1), & \text{if } \frac{\delta E}{\delta w_{ij}}(t-1) * \frac{\delta E}{\delta w_{ij}}(t) > 0 \\ \eta^{-} * \Delta_{ij}(t-1), & \text{if } \frac{\delta E}{\delta w_{ij}}(t-1) * \frac{\delta E}{\delta w_{ij}}(t) < 0 \\ \eta^{0} * \Delta_{ij}(t-1), & \text{otherwise} \end{cases} \tag{4.7}$$

where $\eta^0 = 1$, $0 < \eta^- < 1 < \eta^+$ and $\eta^{-,0,+}$ are known as the update factors and $*$ denotes the multiplication operand. Whenever the derivative of the corresponding weight changes its sign, it implies that the previous update value is too large and it has skipped a minimum. Therefore, the update value is then reduced (η^-) as shown above. However, if the derivative retains its sign, the update value is increased (η^+). This will help to accelerate convergence in shallow areas. To avoid over-acceleration, in the epoch following the application of η^+, the new update value is neither increased nor decreased (η^0) from the previous one. Note that values of Δ_{ij} remain non-negative in every epoch.

This update value adaptation process is then followed by the actual weight update process, which is governed by the following equations:

$$\Delta w_{ij}(t) = \begin{cases} -\Delta_{ij}(t), & \text{if } \frac{\delta E}{\delta w_{ij}}(t) > 0 \\ +\Delta_{ij}(t), & \text{if } \frac{\delta E}{\delta w_{ij}}(t) < 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.8}$$

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t) \tag{4.9}$$

4.2 Radial Basis Networks

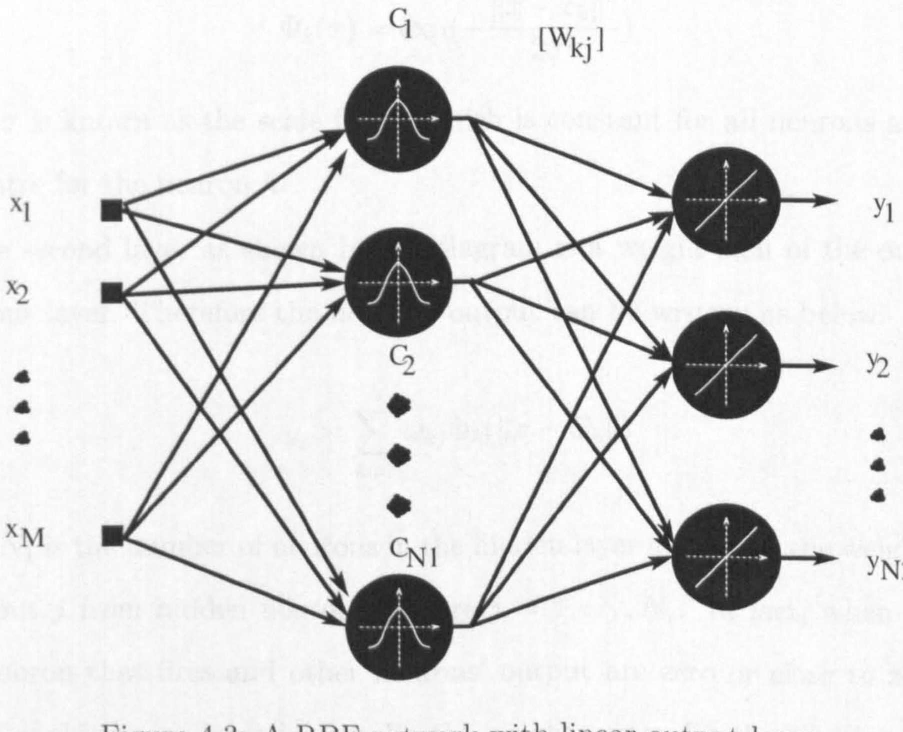


Figure 4.3: A RBF network with linear output layer

Another popular type of feed-forward ANN is the radial basis function (RBF) networks [56, 57]. RBF networks would normally have more hidden layer neurons than MLPs. In fact, a typical RBF has one neuron for every training examples. However, there also exists implementation where the hidden layer neurons are added one at a time. To do this, the training pattern that produces smallest training error will be chosen as the new hidden layer neuron.

Typically, RBF is a two layer network as shown in Figure 4.3. The activation function for the input layer of RBF is:

$$f(x) = \Phi(\|x\|) \tag{4.10}$$

where $\Phi(\cdot)$ is a continuous function (a radial basis function). The activation function can take the form of a Gaussian activation function, thus for k^{th} neuron in the hidden layer:

$$\Phi_k(x) = \exp\left(-\frac{\|x - c_k\|^2}{2\sigma^2}\right) \tag{4.11}$$

where σ is known as the scale factor which is constant for all neurons and c_k is the centre for the neuron k.

The second layer as shown in the diagram is a weight sum of the output of the input layer. Therefore the network output can be written as below:

$$y_j = \sum_{k=1}^{N} \omega_{kj} \Phi_k(\|x - C_k\|) \tag{4.12}$$

where N_1 is the number of neurons in the hidden layer and ω_{kj} is the weight value of output j from hidden neuron k where $j = 1, \ldots, N_2$. In fact, when there is only neuron that fires and other neurons' output are zero or close to zero, the output of this linear layer is actually the weight vector for the winning neuron. However, this is an extreme case, normally, all neurons would have a certain degree of response.

One way to see the difference between a MLP and a RBF network is to look at the way the decision boundaries are drawn. In MLP, the decision boundary are drawn as hyperplanes. However, in RBF networks, decision boundaries exist as countours in the feature space. A comprehensive illustration of this can be found in [58, Chapter 3].

4.2.1 Learning Rule for RBF

The RBF networks have two stages for learning, an unsupervised stage for the centres and a supervised stage for the weights update.

In the unsupervised stage, each time an input vector, $x(n)$, is presented to the network,the distances between $x(n)$ and the centres $C_i(n)$ are computed. The winning centre,C^{win} is then the one with the smallest distance, D^{win}:

$$D^{\text{win}}(n) = \arg\min_{D} D_i(n) \tag{4.13}$$

The winning centre is then updated according to the following:

$$C^{\text{win}}(n+1) = C^{\text{win}}(n) + \mu(x(n) - C^{\text{win}}(n)) \tag{4.14}$$

where μ is a small positive constant analogous to the learning rate parameter in MLP. After the unsupervised stage, the weights update of the output layer is then carried out. If the output layer is linear one, then the Least Mean Squared (LMS) algorithm can be used. For each iteration, the errors between the targets (desired output values) to the network's outputs are found. Consequently the weights are updated according to:

$$w_{kj}(n+1) = w_{kj}(n) + \alpha e_j(n)\Phi_k(\|x - c_k\|) \tag{4.15}$$

where again α is a small positive constant and $e_j(n)$ is a small. If the output activation function is not a linear one, then standard BP algorithm can be applied in place of the LMS algorithm.

4.3 Probabilistic Neural Networks

Another family of neural networks, which is structurally similar to the RBF networks, is the probabilistic neural networks (PNN). The PNN is in fact an out growth from Bayesian classifiers and evolved from a classifying method named Parzen's windows. Historically, the PNN was first proposed for pattern classification by Specht in 1967 [59] and since then had been used in various applications e.g. vector-cardiogram interpretation [60], power lines fault detection [61], spike detection for on-line vibration diagnostics [62], etc. However, the PNN was not a straight hit off during the early days of its introduction due to the computational constraint which limited its usage in real time or dedicated application. Nevertheless, with the massive advances in computing abilities and parallel computers, the PNN has enjoyed renewed interest [63, 64] and a few modified version to were proposed [65, 66] recently.

The PNN has some advantages over MLP and other BP methods. Firstly, the PNN has a rapid training time when compared to BP. The iterative training in BP could take a long period of time, but this is replaced by a little more than reading in the training set in the case of PNN. Secondly, PNN is said to be guaranteed to converge to a Bayesian classifier given enough training examples. Such guarantee can not be found with standard BP procedures, and as a matter of fact, standard BP has the risks of become trapped in the local minimums of the cost functions. Additionally, the PNN also allows convenient data alteration, i.e. pattern can be added or deleted from the training set without lengthy retraining. This means that when there are more data available as the time goes by, we can add to the PNN so that better classification can be made. One other advantage over the MLP is that the output of PNN indicates the confidence of the decision. Such a level of evidence cannot be found in MLP.

Although the PNN overcomes some major shortfalls of MLP, it still retains the desired characteristics of neural networks. Like MLP, it is able to map complex input/output relation via learning. It is also able to generalise data, i.e. if a similar pattern is found, it is able to classify the new pattern correctly within limits. Last but not least, it also retains the concurrency property of neural networks enabling parallel implementation for real time processes.

4.3.1 Bayesian Classification and Parzen Windows Estimation

The Bayesian method of N class classification chooses class a with the feature value, x, which maximises product of the prior probability P_a, the probability $p_a(x)$, and its loss factor, $l_a(x)$ of the class a, $a = 1, 2, \ldots, N$. In its simplest case, if the prior probabilities and the loss factors are equal for all classes, the Bayesian classifier simply chooses the class with the greatest probability value for feature, x to be observed. This can be illustrated in a dual class scenario using Figure 4.4. In Figure 4.4, the one dimensional Normal PDFs of two classes, Class 1 and Class 2, are shown for one dimensional input feature, x. Given an observed value of $x = 0.2$ as shown with the vertical dotted line, $p_1(x)$ gives a value of 0.04 and $p_2(x)$ give a value of 0.77. The decision of Bayesian Classifier in this case therefore is that the observation belongs to Class 2. Of course, in practical scenario, there are often more than one dimensional input spaces, and hence a multi-dimensional PDF is used for such decision making processes.

In real life, the PDF for each class is general unknown and has to be deduced from the training samples. It is often difficult to estimate the PDF accurately

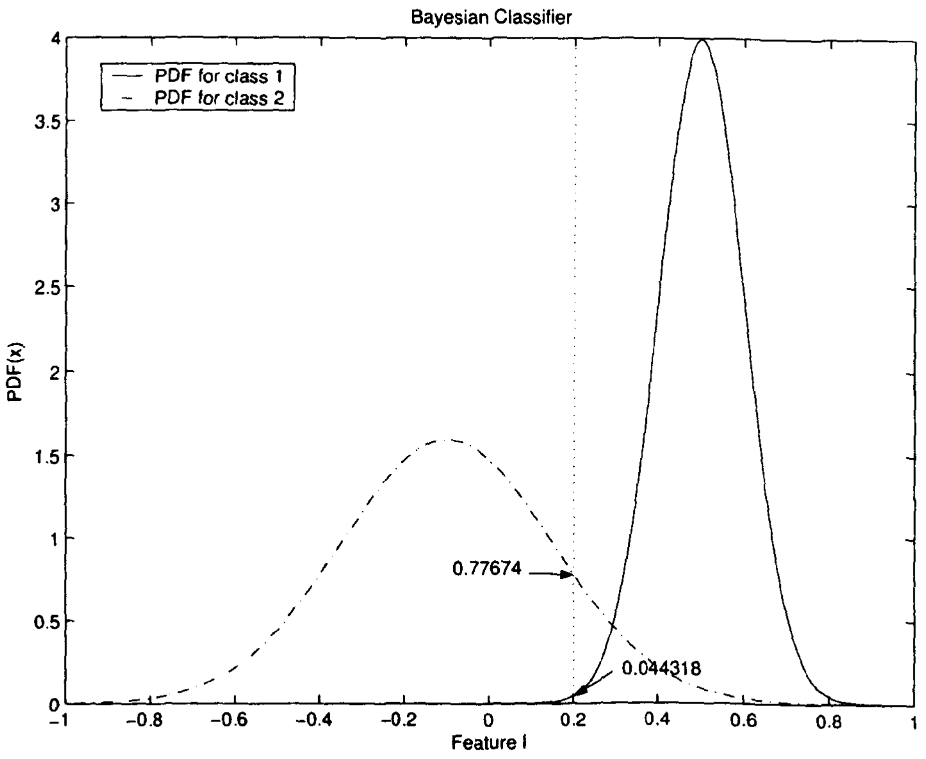


Figure 4.4: Illustration of Bayesian classification

because the training data is often quite sparse in nature, therefore not allowing sufficient density for the histogram of the training data to be meaningful. Parzen proposed a method in overcoming such problem in 1962 and this method is now generally called Parzen windows [67]. Figure 4.5 shows the method is used for one feature and one class. Three samples and their unit Gaussians are shown in dotted lines in Figure 4.5. Adding them up together with scaling, an estimation of the true PDF is shown with the thicker solid line. Parzen showed that with large number of samples and appropriate scaling, this composite curve will approach the true PDF.

The calculation of whole PDF is not necessary while implementing Parzen's windows for classification. One only need the value of Parzen's windows at the test vector point, which is summarised in the following equation, generalised for

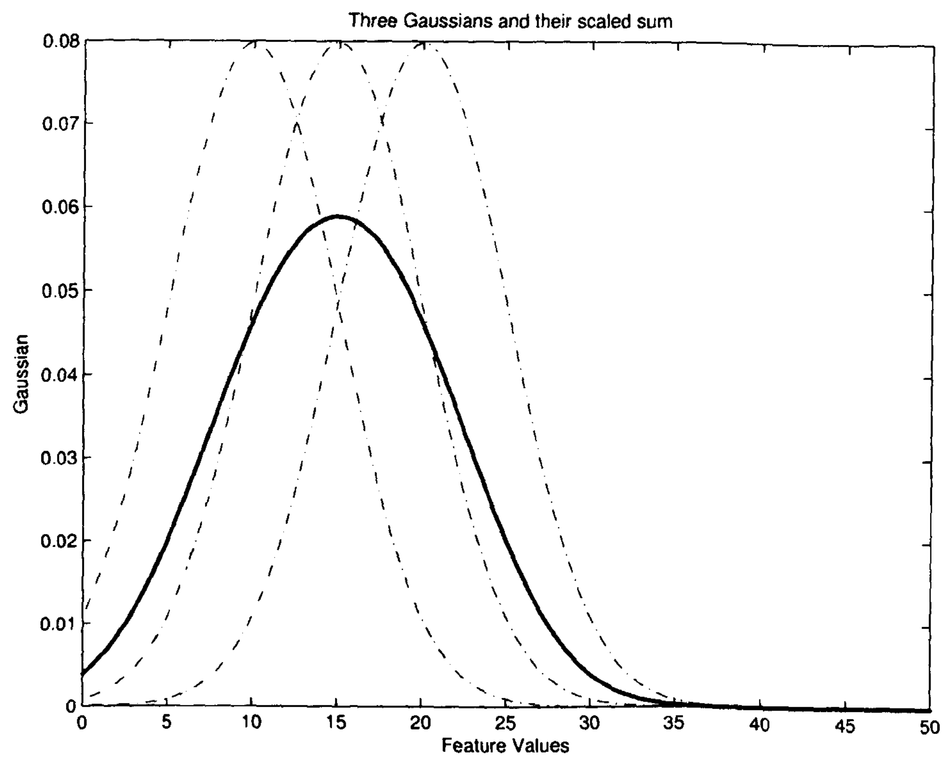


Figure 4.5: Parzen's windows of PDF estimation

d-dimensional case:

$$p_a(\mathbf{X}) = \frac{1}{(2\pi)^{\frac{d}{2}} \cdot \sigma^d \cdot n_a} \sum_{i=1}^{n_a} \exp\left(-\frac{(\mathbf{X} - \mathbf{Y}_a^i)^t(\mathbf{X} - \mathbf{Y}_a^i)}{2\sigma^2}\right) \tag{4.16}$$

where i is the sample number, d is the number of dimension of the feature space, σ is the scale factor as with the case of RBF networks, $\mathbf{X}$ is the test vector to be classified, and Y_a^i is the i^{th} training vector of class a. When the scale factor σ, approaches zero, the classifier then approximates a nearest neighbour classifier. When σ approaches to infinity, the decision boundaries reduces to a hyperplane, therefore only linearly separable problems can be solved.

The popularity of neural networks in 1980s led to the reformulation of Parzen's windows with neural networks terminology. The idea of Parzen's windows can be recreated using a PNN with a three-layer network. These layers include a pattern layer, a summation layer and an output layer. Structurally the PNN is similar to the RBF networks, except the output layer can be now called a decision layer since it is replaced with competitive layer and therefore only one neuron (representing one particular class) can fire at one time.

More detailed discussion on MLP, RBF and PNN can be found in [68, 69] and other neural network related literatures. A bibliography of application of neural networks in digital communications can also be found in [52].

4.4 Experimental Setup and Results

In this section, we investigated and examined the three neural networks that were discussed previously. For the purpose of the experiments in this work, some statistical feature sets (Equation (3.16)) were generated with a variety of selected parameters. The datasets were generated according the parameters shown in Table 4.1. These features were then generated and grouped according to their

Parameters	**Varieties**
Modulation	ASK2, BPSK, QPSK, PSK8, V.27, V.29, Star(V.29c), V.32, QAM8, QAM16, QAM32 & QAM64
SNR	-5 dB, 0 dB, 5 dB, 10 dB, 15 dB & 20 dB
Total Number of Symbols	100, 250 & 500

Table 4.1: Parameters for feature sets

SNRs and total number of symbols from which the three statistical features were generated. For each modulation type, 3000 examples for each SNR and each setting of total number of symbols were generated. These 3000 examples were then divided into 3 equal set of a thousand for training, validation and testing. This high number of training and test examples ensured the statistics for classification were as meaningful as possible.

For the following sections, we first looked at the determination of hidden neurons for the case MLP with BP and RPROP. The BP that was adopted here incorporates learning with momentum and early stopping. Then, the asymptotic behaviour of the feature sets was examined by comparing the effect of the estimation of features using different number of symbols. The performance of RBF and PNN with a variety of scale factors is compared with the best performance obtained using RPROP classifier.

The general model for a ANN based classifier is shown in Figure 4.1. From the diagram, it implicitly shows that the number of input nodes for the ANN classifiers is determined by the number of input features; in this case, three. The number of output neurons is then determined by the number of available classes. It has been shown that a two-layer network (a hidden layer and output layer) has the ability to approximate any measurable function in a precise fashion provided there are sufficient number of hidden layer neurons [70, 71]. Therefore, a two-layer network is adopted in this work.

Under this kind of setup, one now needs to find the optimal number of hidden layer neurons for MLP based classifiers. A general rule of thumb is that the hidden layer neurons is at least twice the input features. For RBF and PNN based classifiers, the number of hidden layer neurons are assigned implicitly in the training algorithm.

4.4.1 Case Study I: SNR Analysis and Hidden Layer Neurons

In order to find out the optimal (or at least suboptimal) number of neurons, two computer experiments were conducted in MATLAB, in conjunction with the Neural Network Toolbox. For each experiment, a set of two-layer MLP feed-forward NNs was created with different number of hidden layer neurons, ranging from one neuron to a layer of twenty neurons. Two training algorithm were then used for investigation namely, BP learning with momentum and adaptive learning rate, and the RPROP algorithm. For both algorithms, the nonlinearity for the hidden layer neurons was set to the *tansig* (sigmoid) function, while the nonlinearity of the output layer neurons was set to *logsig* (logarithmic sigmoid) function[1]. For each algorithms, SNR analyses were conducted with each setting of number of hidden layer neurons.

At the end of each training iteration, the network under test was tested with all the three sets of data, giving three set of performance, which were then averaged to give the performance index[2].

Equation (4.17) was used as performance index for this section:

$$P_{avg} = \frac{P_{\text{training}} + P_{\text{validation}} + P_{\text{testing}}}{3} \tag{4.17}$$

where P_{training}, $P_{\text{validation}}$ and P_{testing} are the correct classification percentages of

[1]The *logsig* function has the range from 0 to 1 that corresponds to the range of target vectors ($N \times 1$ vectors with a single entry of '1' at the row corresponding to the class label, and '0' elsewhere).

[2]It is not common practise to use such a performance index; if the classifier are over-fitted to the training data set, this index will be bias to a higher value. However, as seen from Figure 4.6, all three data sets show closely matched and well generalised performance. Therefore, in this case, the usage of such index can be compromised. A more common approach is to use the classification performacne of the test set as the performance index, which will be adopted in the author's future works.

training, validation and testing process. Figure 4.6 shows a typical graph to depict a typical training process. As can be seen from the graphs, the network was generalising well as there were little differences in the MSE of each curves.

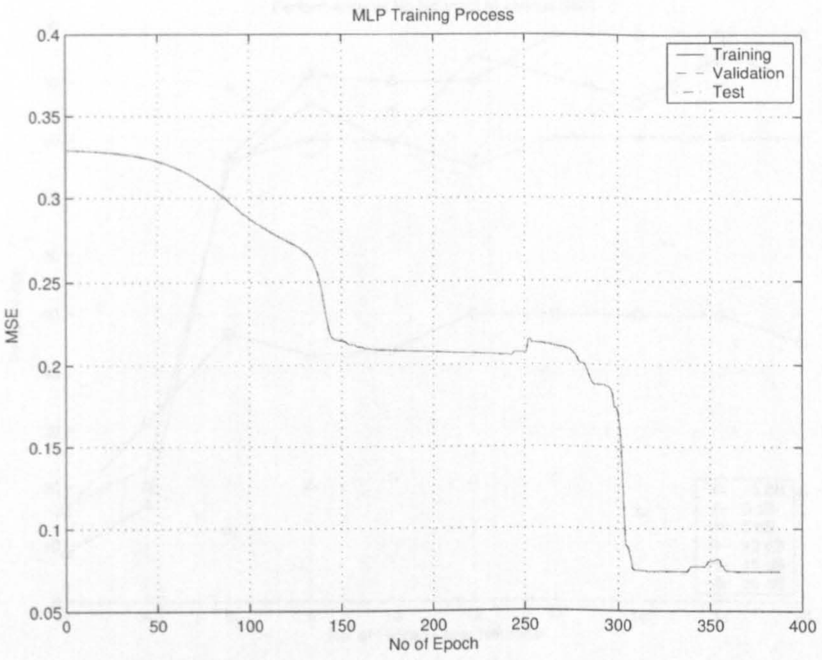
Figure 4.7(a) gives the result obtained from the experiment using BP learning with momentum and adaptive learning rate. The horizontal axis gives the number of neuron and the vertical axis gives the performance index of the classifier. Results for SNR $= -5, 0, 5, 10, 15,$ and 20 dB are shown using different markers as depicted in the legend box.

The results obtained for this experiment did not clearly indicate the optimal number of neurons. From the graph, it did show a general trend of convergence once a certain number of neurons was achieved for all cases with different SNRs. Ideally, the performance of the classifier should converge if the classifier possesses sufficient neurons to draw the decision boundaries.

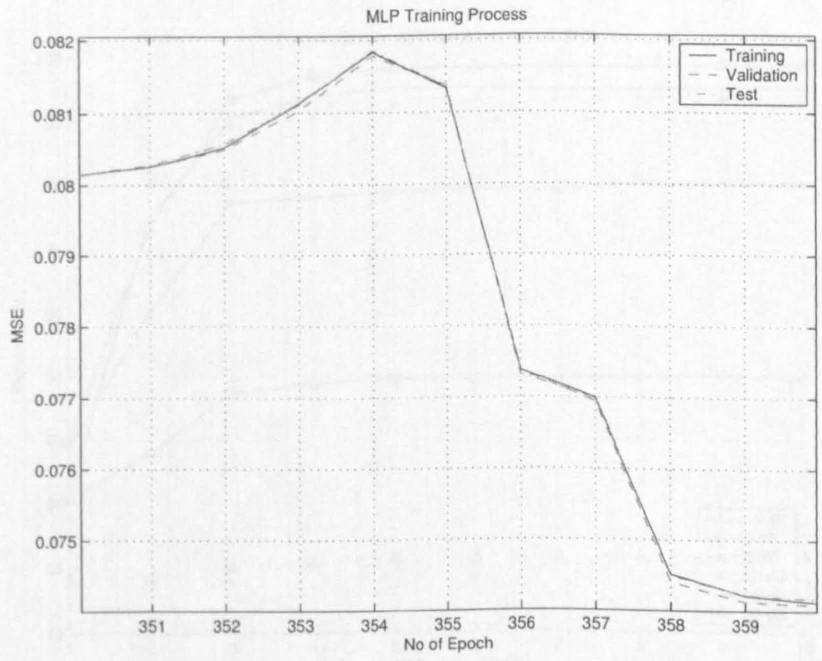
The number of hidden layer neurons chosen was fourteen in this case. The reason being, with fourteen neurons, the performance of most classifier had adequately converged, particular for SNR = 20 dB and 5 dB. The fluctuation of performance at lower SNR seemed to be less prominent than with higher SNRs.

With the RPROP classifier, the picture was much clearer than it was with the case of BP training. In Figure 4.7(b), the results for the RPROP classifier is shown using identical setup as with BP classifier, with the exception that, the horizontal axis, representing number of hidden layer neurons, has been truncated to 10 neurons. The six curves again show the results obtained for six different SNR value as depicted in the legendary box.

From the graph, it was shown that the RPROP classifiers were superior to their BP counterparts. All RPROP classifiers gave a converged performance index value with number of hidden layer neurons set to five or more. This was a useful result that one can immediately determine the number of neurons needed for the

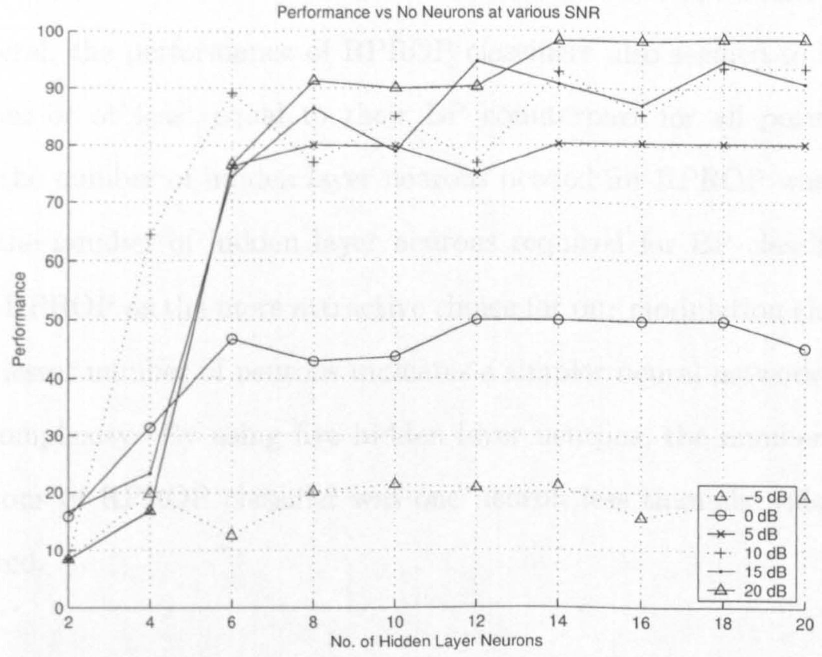


(a) 4 Hidden Neurons with BP at 20 dB SNR

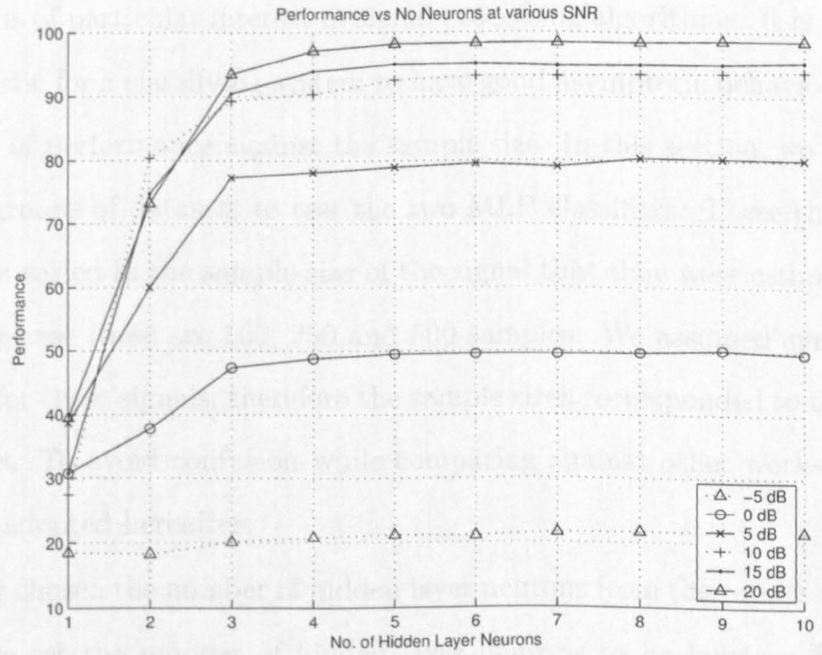


(b) Zoom-in of the above from Epoch 350 to Epoch 360

Figure 4.6: An example of training process for BP learning with momentum and adaptive learning rate



(a) Determining the number of hidden layer neuron required for MLP using BP with momentum and adaptive learning rate



(b) Determining the number of hidden layer neuron required for MLP using RPROP algorithm

Figure 4.7: SNR analyses for the two MLP learning algorithm

hidden layer without much further analysis or compromise to be made.

In general, the performance of RPROP classifiers also seemed to be consistently higher or at least equal to their BP counterpart for all positive SNRs. However, the number of hidden layer neurons needed for RPROP was half compared to the number of hidden layer neurons required for BP classifiers. This suggested RPROP as the more attractive choice for our modulation classification task, as a lesser number of neurons indicates a simpler neural network in term of network complexity. By using five hidden layer neurons, the number of hidden layer neurons of RPROP classifier was one neuron less than the rule of thumb had required.

4.4.2 Case Study II: Asymptotic Behaviour

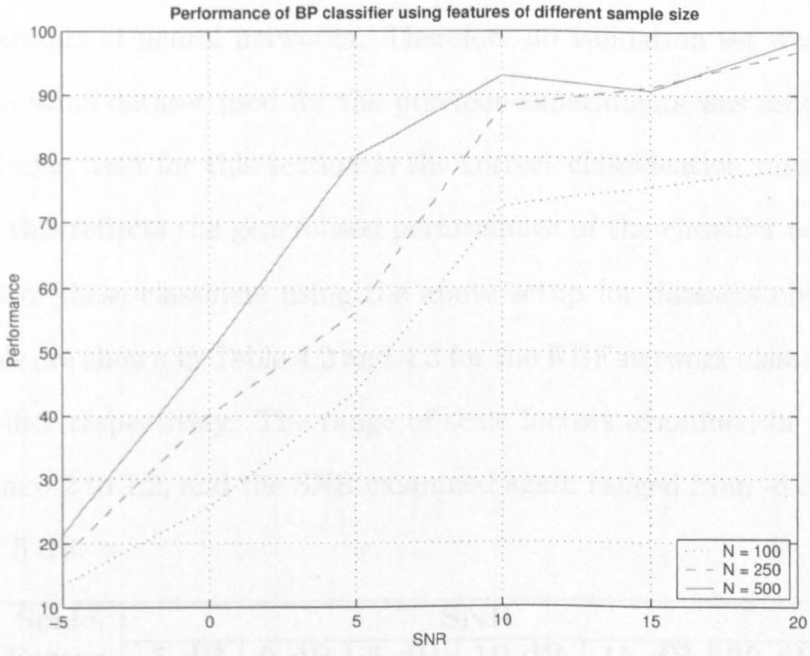
Having designed a feature set, its relation with the sample size that it was estimated from is of particular interest in signal processing algorithms. It is a desirable characteristic for a classifying system to have good asymptotic behaviour, i.e. the increment of performance against the sample size. In this section, we used three identical groups of datasets to test the two MLP classifiers. These three groups of datasets varied in the sample size of the signal that they were estimated from, and in our case these are 100, 250 and 500 samples. We assumed symbol space sampling for these signals, therefore the sample sizes corresponded to the number of symbols. To avoid confusion while comparing against other works, the term symbol is adopted hereafter.

Having chosen the number of hidden layer neurons from the results of previous section, we set the number of hidden layer neurons to be fourteen for the BP classifiers, and five for the RPROP classifiers. Consequently, the classifier were trained for different SNRs as before. The number of maximum training epoch allowed for each classifier was set to 10,000. This will give more chances for the classifier to converge fully. The results obtained are summarised in Figure 4.8(a) and Figure 4.8(b) for the BP classifiers and RPROP classifiers respectively.

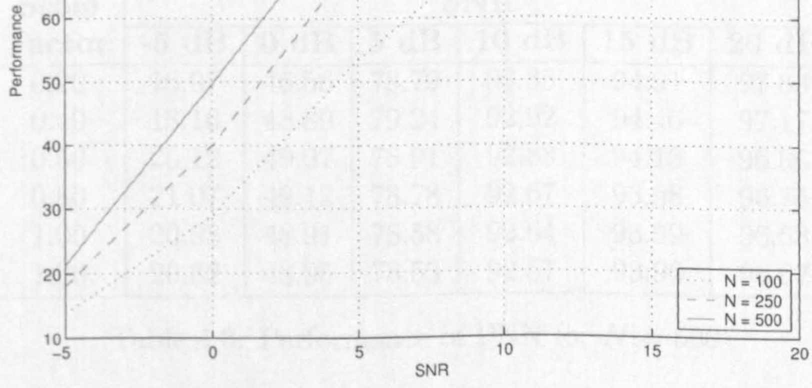
From the figures, both classifiers indicate that a higher sample size dataset would give a better classification performance. In other word, our classifying systems do converge with the increment in sample size. From the results, it can be seen that the RPROP classifier outperformed the BP classifier again, and especially in datasets of low sample size. Interestingly, the performance of BP at 100 symbols and 250 symbols were lower than RPROP initially (0 dB and 5 dB), then over the range of SNR, the performance of the algorithms seemed to converged and leveled at 10 dB and above. At 500 symbols, the BP's performance at 15 dB dropped back to 100 symbols, this could be due to a local minimum encounter in the algorithm.

4.4.3 Case Study III: RBF network and PNN

In this section, we examine the performance of alternative classifiers, based on the RBF networks and PNNs. For these experiments, the training dataset was reduced to one tenth of those used for MLP classifiers. Although more training data was available, the computational burdens of these neural network variants were too much. From experiments, it was deduced that the physical memory was a limitation for the algorithm to train with full dataset. Therefore, the classifier was only shown 100 training examples per modulation type as opposed to 1000 examples per modulation in previous sections. An alternative method that could be used is to train the networks multiple times if one insists to have all the examples used for training. However, this was not adopted and proved unnecessary from the experimental results obtained.



(a) Performance of BP Classifier (using 14 hidden layer neurons) with feature set estimated from different number of symbols



(b) Performance of RPROP classifier (using 5 hidden layer neurons) with feature set estimated from different number of symbols

Unlike MLP neural networks, generalisation was controlled by the scale factor in these variants of neural networks. Therefore no validation set was used. For testing, the same dataset used for the previous experiments was used. The performance index used for this section is the correct classification rate of the test dataset as this reflects the generalised performance of the classifier as well.

Results of these classifiers using the above setup for datasets obtained with 500 symbols are shown in Table 4.2 and 4.3 for the RBF network classifier and the PNN classifier respectively. The range of scale factors examined in this section ranged from 0.2 to 1.2, and the SNR examined again ranged from -5 dB to 20 dB in steps of 5 dB.

Scale	**SNR**					
Factor	**-5 dB**	**0 dB**	**5 dB**	**10 dB**	**15 dB**	**20 dB**
0.20	8.31	9.03	57.79	71.60	67.45	95.85
0.40	8.30	9.69	31.98	32.24	92.74	96.97
0.60	8.97	9.42	25.94	61.08	84.96	92.85
0.80	9.57	14.63	46.77	77.30	90.99	98.19
1.00	9.06	20.11	58.75	84.80	92.62	97.16
1.20	10.25	25.82	65.85	88.10	94.07	98.28

Table 4.2: Performance of RBF network for $N = 500$

Scale	**SNR**					
Factor	**-5 dB**	**0 dB**	**5 dB**	**10 dB**	**15 dB**	**20 dB**
0.20	16.07	46.56	78.79	93.35	94.91	97.83
0.40	18.16	48.89	79.24	92.92	94.46	97.17
0.60	20.12	49.07	78.91	92.83	94.13	96.86
0.80	21.07	49.12	78.78	92.67	93.98	96.75
1.00	20.98	48.91	78.58	92.64	93.99	96.68
1.20	20.82	48.95	78.53	92.57	93.96	96.67

Table 4.3: Performance of PNN for $N = 500$

From Table 4.2, it was found that the best performing scale factor of those examined was 1.2 for the RBF network classifier. At high SNR (10, 15 and 20 dB), the RBF network classifier achieved very similar performance of the RPROP classifier with 5 hidden layer neurons (see Figure 4.8(b)). However, at lower SNRs (-5,0 and 5 dB) the performance of the RPROP classifier was twice as good as the figures shown by the RBF network classifier.

With the PNN classifier, the determination of scale factor was a more difficult task. The optimal scale factor seems to decrease with the improvement of SNR. From the table, it seemed that for -5 dB and 0 dB, the best scale factor was 0.8. For 5 dB and 10 dB, it was 0.4 and for high SNRs, the best scale factor was chosen as 0.2. Nevertheless, for a practical system, we would have to make some trade off, therefore we have chosen the scale factor to be 0.4 after various considerations. Nevertheless, these knowledge is useful if the knowledge of SNR is available. It also prompt for further work in a adaptive version of PNN with variable scale factor.

In comparison with the RPROP classifier, the performance of the PNN classifier showed (scale factor = 0.4) a close match although typical 1 or 2% lower than the RPROP classifier. This is an encouraging result as the training examples used were one tenth of those shown to the RPROP classifier, The results suggested that the possibility of better performance if repetitive training were used for the PNN classifier. One remark from the observation in terms of physical training time, the PNN classifier was the fastest in all the variants of NN investigated in this work. However, the memory requirement for a PNN is much larger than the memory requirement of a MLP. This is because the PNN reads in every training vectors as its patterns.

The experiments were repeated for datasets obtained with 100 and 250 symbols as with the MLP classifiers. The corresponding results were shown in Table 4.4 to Table 4.7. Generally, the results show an increase of performance with increment of the original signal sample size. This is coherent with the observation in Section 4.4.2 in the case of MLP classifiers. This is expected in statistical features as such features are sample estimates of the true statistics of the sampled signals.

Scale	**SNR**					
Factor	**-5 dB**	**0 dB**	**5 dB**	**10 dB**	**15 dB**	**20 dB**
0.20	8.26	11.28	39.52	9.66	51.92	89.74
0.40	8.35	9.40	9.61	15.82	52.73	74.53
0.60	8.14	8.51	12.75	39.85	73.03	85.76
0.80	9.29	11.84	31.38	58.77	80.33	96.22
1.00	9.07	14.87	39.58	69.92	83.94	96.23
1.20	9.21	19.27	47.33	77.47	86.10	94.81

Table 4.4: Performance of RBF network for $N = 250$

Scale	**SNR**					
Factor	**-5 dB**	**0 dB**	**5 dB**	**10 dB**	**15 dB**	**20 dB**
0.20	8.38	8.72	16.62	11.29	39.67	77.81
0.40	8.18	8.58	24.77	14.19	31.85	76.72
0.60	8.16	8.92	11.91	20.38	47.35	64.17
0.80	8.45	9.30	17.97	33.89	57.64	74.81
1.00	8.93	11.79	25.10	43.83	64.24	87.30
1.20	9.07	13.82	29.70	51.67	69.11	88.33

Table 4.5: Performance of RBF network for $N = 100$

From the tables, the same phenomenon that was observed for the scale factors in the case of the RBF networks and the PNN were again observed for 100 and 250 symbols. The RBF network favoured a consistent scale factor (1.2 in this case) but the scale factors for the PNN seemed to get smaller as the SNR increases.

4.5 Conclusion

In this chapter, the general framework of pattern recognition using neural networks was introduced and discussed. Three popular types of neural network were

Scale	**SNR**					
Factor	**-5 dB**	**0 dB**	**5 dB**	**10 dB**	**15 dB**	**20 dB**
0.20	13.38	34.60	68.20	86.77	90.58	96.00
0.40	15.65	38.48	69.87	85.53	89.71	95.51
0.60	16.56	39.38	69.68	84.84	89.34	95.32
0.80	17.00	39.33	69.21	84.38	89.09	95.20
1.00	17.27	39.05	69.07	84.12	89.00	95.18
1.20	17.55	38.86	68.85	83.90	88.97	95.15

Table 4.6: Performance of PNN for $N = 250$

Scale	**SNR**					
Factor	**-5 dB**	**0 dB**	**5 dB**	**10 dB**	**15 dB**	**20 dB**
0.20	9.83	22.40	48.36	69.28	77.86	89.49
0.40	11.50	26.47	52.44	69.87	77.17	88.33
0.60	12.85	27.54	52.77	69.20	76.54	87.30
0.80	13.30	27.84	52.62	68.54	76.04	86.83
1.00	13.23	28.11	52.26	68.28	75.70	86.61
1.20	13.35	28.02	51.87	67.95	75.53	86.54

Table 4.7: Performance of PNN for $N = 100$

investigated and their performance was examined using the statistical features proposed in previous chapter. Selected results were included at the end of the chapter.

The first neural network variant investigated was the MLP neural network. With a feed forward structure and structure resemblance of the physical neural network, this variant gained huge popularity since the 80s particular among the new comer into field. Two training algorithms were investigated for the case of MLP classifiers, these were namely the standard BP with momentum and adaptive learning rate, and the RPROP algorithm. In terms of performance, the RPROP classifier consistently outperformed the BP classifier. From observation, the RPROP classifier also showed faster convergence during training. The optimal number of hidden layer neurons for the classifiers were chosen via Monte Carlo trials for a range of numbers of hidden layer neurons.

The asymptotic characteristic of the features proposed were also examined through experiments with the two chosen classifiers with an optimal number of hidden layer neurons. It was shown that the the higher the original sample size, the better the sample estimates, and hence the better the performance shown. This observation was later confirmed with a coherent result for other variants of neural networks investigated in this chapter.

In the final part of the chapter, the performance of RBF network classifier and the performance of PNN classifier were investigated through computer experiments. The scale factors for the networks, although different in their origins, played an important roles for the generalisation for the neural networks. The PNN in particular was shown to be a robust alternative to MLP based solution, boasting implicit generalisation, faster training time and no threats of trapping in local minimum of cost functions. Performance wise, the PNN closely matched the RPROP classifier, while RBF network classifier lagged behind to the two other alternatives in lower SNRs.

In short, supervised neural networks provide an easy to implement and theoretically easy to comprehend approach to modulation classification. With the 12 modulation types tested in this work, all three neural networks variants gave excellent performance of near 100% in simultaneous learning/classification of all 12 modulation types. As far as the knowledge of the author is concerned, none of the other previous works using ANN reported such a broad range of digital modulation types. Unfortunately, direct comparison of results with other work is difficult because of different modulation types and signal datasets used in different works (as agreed by [29]). Furthermore, the existence of various definitions for SNR also complicate the task of result comparison. Nevertheless, supervised neural networks give a robust alternative solution to the decision theorectic approach in digital classification.

Chapter 5

On Feature Selection and Transformation

Feature design as discussed in Chapter 3 has always been a crucial and integral process of any pattern recognition system. An excessive number of features as input to any classification system is undesirable as it induces more computational overhead and requires more training samples as was discussed earlier. The technique of choosing the right set of feature subset without compromising on performance is often called *feature subset selection.*

Two approaches of feature subset selection are investigated here in this work, i.e. feature selection and feature transformation. Feature selection iteratively selects a combination of features and evaluates the performance of the selected feature subset. This is often carried out in an organised fashion, e.g. forward selection, backward selection, etc. This type of feature subset selection method is often called the wrapper type method as the classifier is used as the evaluating agent in the algorithm. The main drawback of the wrapper type method is that these algorithms often take a long time to complete and this is may be a factor to consider for those who require a solution quickly.

The other approach, feature transformation, exploits some pre-defined inter-relationship of the features and applies a linear projection of the original feature set to create a new feature sets (often with less dimensionality). Feature transformation is more often used for feature extraction; however, it is demonstrated that it can be used as a feature subset selection method. In terms of evaluating functions, feature transformation would be grouped under the approach called filter type feature selection, as the classifier is not involved in choosing the new feature subset. However, it differs from the main stream filter type method as each of the new features is now a combination of a group of features from the original feature set. Although, in principle, the transformation method is quicker than the selection method, the performance of the chosen feature is not guaranteed in terms of classification as the actual classification does not take part in the selection process.

The RPROP-MLP classifier is chosen as the classifier throughout this chapter to enable performance comparison between different methods.

5.1 Feature Sets

Two datasets are used in this work. The first set is the spectral feature set, which was introduced in Azzouz and Nandi's work in digital modulation classification [17]. The second set is the statistical set, which includes all 12 statistical features (defined by Equation (3.12)) prior to self-normalisation and processing.

For completeness, the spectral feature set are listed in the following section:

5.1.1 Spectral Feature Sets

The main motivation for this spectral feature set is that, the information content for digital modulations is hidden either in the signal instantaneous amplitude, instantaneous phase or instantaneous frequency, while for the previously proposed modulation types [17, chapter 3], information is only hidden in a single domain for a particular modulation type.

The five features proposed previously are as below:

• Maximum value of the power spectral density of the normalised-centred instantaneous amplitude:

$$\gamma_{max} = \arg\max \frac{1}{N_s} DFT(a_{cn}(i))^2 \tag{5.1}$$

where N_s is the number of samples, $a_{cn}(i) = a_n(i) - 1$ and $a_n(i) = \frac{a(i)}{m_a}$, $a(i)$ is the i^{th} instantaneous amplitude and m_a is the sample mean value.

• Standard deviation of the absolute value of the centred non-linear components of the instantaneous phase:

$$\sigma_{ap} = \sqrt{\frac{1}{C}\left(\sum_{a_n(i)>a_t} \phi_{NL}^2(i)\right) - \left(\frac{1}{C}\sum_{a_n(i)>a_t} \|\phi_{NL}(i)\|\right)^2} \tag{5.2}$$

where C is the number of samples in $\{\phi_{NL}(i)\}$ for which $a_n(i) > a_t$ and a_t is the threshold value for $a(i)$ below which the estimation of the instantaneous phase is very noise sensitive.

• Standard deviation of the direct value of the centred non-linear component of the instantaneous phase:

$$\sigma_{dp} = \sqrt{\frac{1}{C}\left(\sum_{a_n(i)>a_t} \phi_{NL}^2(i)\right) - \left(\frac{1}{C}\sum_{a_n(i)>a_t} \phi_{NL}(i)\right)^2} \tag{5.3}$$

• Standard deviation of the absolute value of the normalised-centred instantaneous amplitude:

$$\sigma_{aa} = \sqrt{\frac{1}{N_s}\left(\sum_{i=1}^{N_s} a_{cn}^2(i)\right) - \left(\frac{1}{N_s}\sum_{i=1}^{N_s} \|a_{cn}(i)\|\right)^2} \tag{5.4}$$

• Standard deviation of the absolute value of the normalised-centred instantaneous frequency:

$$\sigma_{af} = \sqrt{\frac{1}{C}\left(\sum_{a_n(i)>a_t} f_N^2(i)\right) - \left(\frac{1}{C}\sum_{a_n(i)>a_t} \|f_N(i)\|\right)^2} \tag{5.5}$$

Note that the availability of instantaneous amplitude, phase and frequency depends on the availability of carrier frequency f_c. It is assumed that f_c is known *a priori*; in practical applications it would need to be estimated. Besides the frequency estimation (see Appendix B in [17]), other concerns such as channel information are assumed to have been adequately addressed either with *a priori* knowledge or through blind estimation algorithms. The above features were used in previous works of Nandi and Azzouz.

5.1.2 General Experimental Setup

Combining the spectral feature set with the 12 statistical features, a total set of 17 features is obtained. For the purpose of examining the feature selection and transformation algorithm, the RPROP MLP classifier is adopted. A range of SNR values is chosen for investigation: -5 dB, 0 dB, 5 dB, 10 dB and 20 dB.

The SNR is again defined as follows:

$$\text{SNR in dB} = 10\log_{10}\left(\frac{\sigma_x^2}{\sigma_n^2}\right) \tag{5.6}$$

where σ_x^2 corresponds to the variance (power) of the the signal, x, and σ_n^2 represents the noise variance (power).

Ten modulation types are used in this work, including frequency shift keying (FSK), which was not examined in the previous chapter. The 10 modulation types are listed in Table 5.1.

Modulation Types	
OOK	ASK4
BPSK	QPSK
FSK2	FSK4
QAM16	V. 29
V. 32	QAM64

Table 5.1: 10 digital modulation types investigated in this chapter

Time series of the these modulation types were created and additive white Gaussian noise was added with a noise factor, R_{sn}, to achieve the desired SNR.

$$R_{sn} = \sqrt{\frac{\sigma_x^2}{\sigma_n^2}10^{\frac{-SNR}{10}}} \tag{5.7}$$

Three datasets were created at each chosen SNR. For each datasets, 1000 examples of each modulation types were created and noise was added accordingly. The three datasets at each SNR were used for training the classifiers, the validation process during classifier training and testing the classifier. These datasets were therefore labeled as the training set, validation set and test set respectively. Table 5.2 shows a set of parameters used in preparation of the datasets.

Parameters	Values
Sampling Frequency, f_s	307.2 kHz
Carrier Frequency, f_c	17 kHz
Baud Rate, R_b	9600 baud
No. Samples, N_s	4096 samples

Table 5.2: Parameters used for digital modulation

5.2 Feature Subset via Selection Method

There exists two main paradigms of feature selection, i.e. the filter type approach (e.g. [72, 73]) and the wrapper type approach (e.g. [74, 75]). For the filter type approach the feature set is evaluated without the aid of the application, in this case, the MLP classifier. In other words, the features were selected based on some predefined functions of the members of the features set. An example of these functions is the correlation among the features. The wrapper type approach, however, uses the performance of the classifier to evaluate the selected feature subset. This has the advantage of guaranteed performance, but often takes a longer time.

There are many methods for feature selection, but they generally consist of two main parts - a selection phase and evaluation phase. In pattern recognition, the selection criterion is normally the minimisation of recognition errors. This leaves us with choosing a suitable selection procedure. In this work we shall focus on a stochastic search method using a genetic algorithm (GA). Similar wrapper type feature selection method with GA were also proposed in [76, 77, 78]. Conventional brute force searching method like forward-selection, backward-selection etc. are also available; a comparison among these methods can be found in [79].

5.2.1 Genetic Algorithm

GA [80, 13] is a stochastic optimisation algorithm which mimics Darwin's theory of survival of the fittest. The algorithm runs through an iteration of individual selection, reproduction and evaluation stages. Each individual represents a unique solution to the problem and for each generation (i.e. epoch in ANN terminalogy), there exists a collection of solutions which is termed a population. The fitness of an individual refers to the quality of the solution which is evaluated at the end of a generation. Prior to evaluation stage, the previous population breeds through reproduction functions, such as crossover and mutation, gives rise to a new population. The new population inherits some preferred properties from the parent population as only individuals with strong fitness value are chosen to reproduce.

1. Initialise a population P_0 of N individuals
2. Set generation counter, $i = 1$
3. Create an intermediate Population, P_i', through selection function
4. Create a current Population, P_i through reproduction functions
5. Evaluate current population
6. Increment the generation counter, $i = i + 1$
7. Repeat step 3 until termination condition reached
8. Output the best solution found

Table 5.3: An overview of generalised GA

Table 5.3 shows a generalised form of the GA. Various advanced GAs exist but all GAs are based on this general concept. At each iteration, or generation in the context of GA, a pool of individuals is created. Each individual is uniquely represented by its genome which is analogous to human genome. The fitness of each individuals is then tested using an evaluation function, which will be discussed later. A pre-defined selection criteria is then used to choose the individuals to reproduce through the genetic operators. The process is then repeated for the next generation. At each generation, the performance is always checked to see whether the GA has converged. Nevertheless, to avoid early convergence, this option is only enabled after a predefined number of generations.

Two important issues in GA are the genetic coding used to define the problem and the evaluation function. Without these, GA is nothing but a meaningless repetition of procedures. The simple GA proposed by Goldberg in his book [13] uses binary coding, but other methods such as real coding are sometimes more meaningful to the problem. This is discussed in further details in the following section 5.2.2.

The fitness function evaluates how good an individual is in surviving the current environment. In function minimisation problems, the individual that gives the smallest output will be given the highest score. In this case, the fitness function can be the reciprocal of output plus a constant. In the modulation classification problem, the evaluation is defined as the overall percentage of correction classification of the classifier. Hence, the GA implemented here is designed to search for the indivuals that represent the classifiers with high accuracy in classfication.

5.2.2 String Representation

Each individual solution in GA is represented by a genome string. This string contains specific parameters to solve the problem. In our application, two different methods of coding are investigated, i.e. the binary genome coding and the list genome coding.

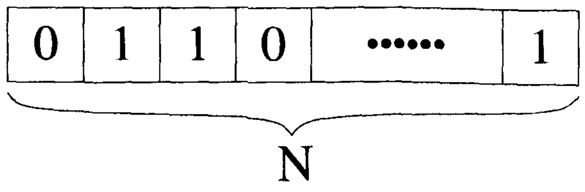


Figure 5.1: An example of binary string genome with length N

In the binary genome coding (see Figure 5.1, each string has a length N, where N is the total number of input features available. A binary '1' denotes the presence of the feature at the corresponding index number. Similarly, a binary '0' denotes an absence. The advantage of this coding is that it searches through the feature subspace dynamically without user defined number of subset features. No constraint is needed with this coding method.

The second coding used is the integer list genome string as shown in Figure 5.2. Each genome in this category is of length M, where M is the desired number of features in a feature subset. To initialise, the GA chooses randomly M integers from a list of integers ranging from 1 to N. However, we do not desire any repetition of the integers as this means that the same feature is selected more than once. Therefore a constraint, $1 < f_i < N$ is applied, where f_i denotes i^{th} input feature. In practice, we randomise the list sequence and choose the first M features in the list.

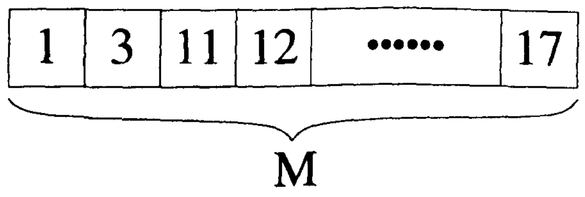


Figure 5.2: An example of integer list string genome with length M

An extra parameter can be appended to the genome for choosing the number of hidden units in the MLP classifier. However, from previous experience [30], it was determined that the optimum number of hidden units is around ten neurons using RPROP training algorithm. Since this is practical and reasonable for a hardware implementation, we shall not complicate our problem of feature selection with an extra parameter.

5.2.3 Basic Genetic Operators

Basic genetic operators are used for reproduction and selection. The following gives a brief description of each operator:

Crossover: Crossover occurs with a crossover probability of P_c. A point is chosen for two strings where their genetic informations are exchanged (Figure 5.3). There are also variations of two-points or multi point crossover. For our purpose, we shall use one-point crossover, and typical value of probability of crossover,$P_{\text{crossover}}$, of 0.75.

Mutation: Mutation is used to avoid local convergence of the GA. In binary coding, it just means the particular bit chosen for mutation is inverted to its complement (Figure 5.4). For the list genome, the chosen index is replaced with a new index without breaking the constraint. Mutation occurs with typical mutation probability, P_{mutation}, of 0.05. This probability value is kept at such a low value to prevent unnecessary oscillation.

Selection: There are several ways to select a new intermediate population. Based on the performance of individual strings, roulette wheel selection assigns a probability to each string according to their performance. Therefore poor genome strings will have a slight chance of survival. Unlike roulette wheel, selection by rank just orders the individuals according to their performance and select copies of best individuals for reproduction.

Other genetic operators, such as elitism, niche and diploidy, are often classified as advanced genetic operators [13]. For the purpose of this investigation, we shall apply only elitism. Elitism comes in various forms. In our application, we require that the best two strings are always to be included in the new population. This gives a chance to reevaluate their capabilities and improves GA convergence.

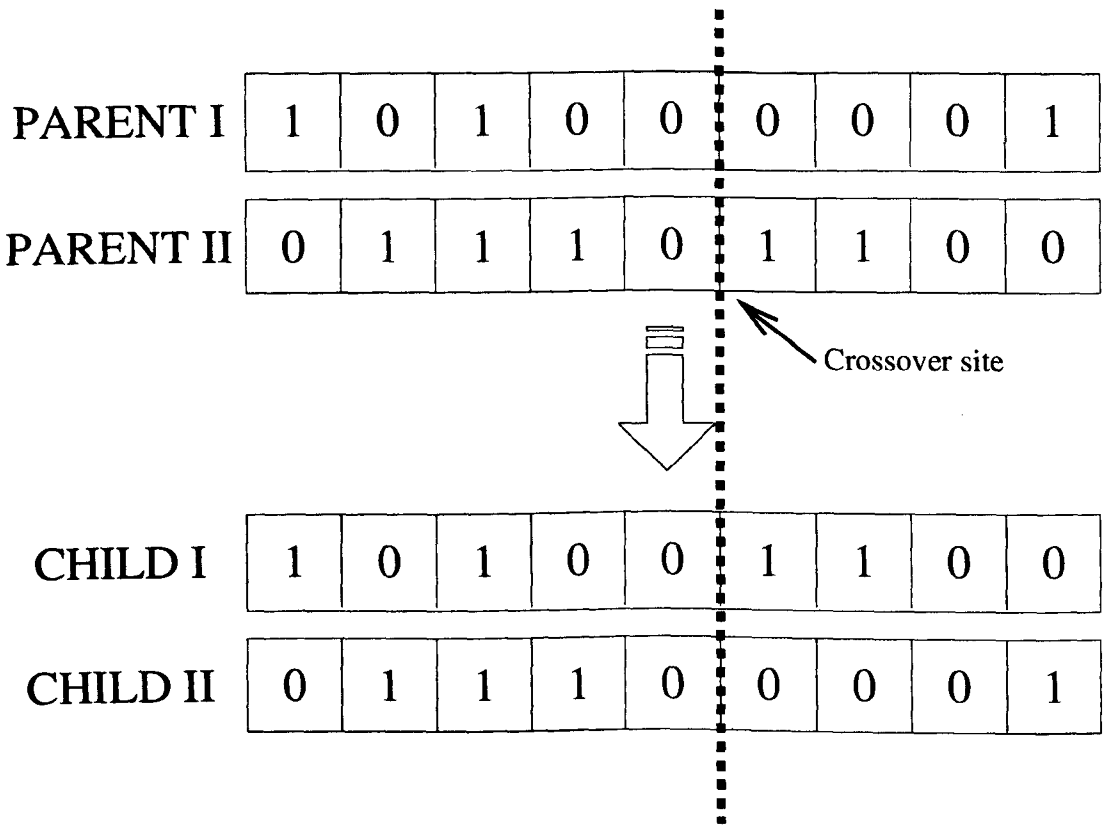


Figure 5.3: Illustration of the single point crossover genetic operation

5.3 Feature Subset via Component Analysis

The second method of choosing the "right" feature subset is related to the wider of field of component analysis. Two of such methods are investigated in this work,

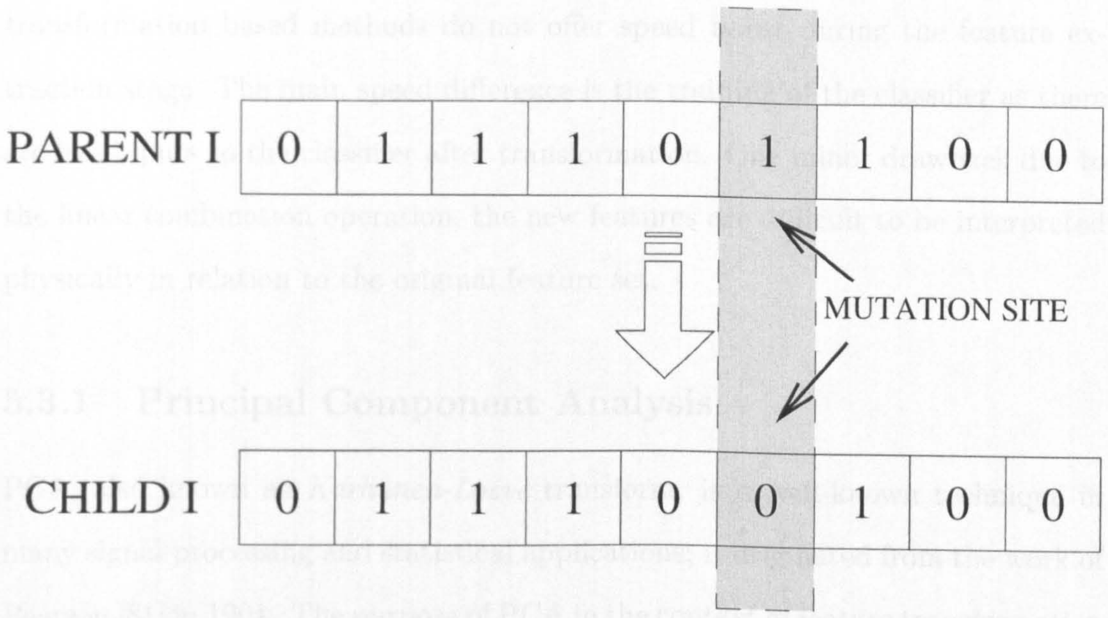


Figure 5.4: Illustration of the mutation genetic operation

these are namely the principal component analysis (PCA) and the independent component analysis (ICA). In term of operation, these methods share the common operation property that both method map the original d-dimensional dataset $\mathbf{F}$ on to a reduced feature set $\mathbf{F}'$ of dimension d' through the mean of linear projection, $\mathbb{L}$.

$$\mathbf{F}' = \mathbb{L}^T \mathbf{F} \tag{5.8}$$

For each method, the linear project, $\mathbb{L}$, is obtain through optimisation of different objective criterion; for example, the PCA seeks the best representation of a dataset according to a minimum-squared-error criterion while ICA seeks features with minimum joint entropy of the data.

The main advantage of these methods is the operational speed in finding the solution relative to selection based methods. However, when used as a tool to select feature subsets, all of the original features have to be present for feature transformation. The reason for this being that the new feature set is a linear combination of the original feature set as mentioned earlier. Therefore, feature transformation based methods do not offer speed boost during the feature extraction stage. The main speed difference is the training of the classifier as there are less inputs to the classifier after transformation. One minor drawback due to the linear combination operation, the new features are difficult to be interpreted physically in relation to the original feature set.

5.3.1 Principal Component Analysis

PCA, also known as *Karhunen-Loéve* transform, is a well-known technique in many signal processing and statistical applications; it originated from the work of Pearson [81] in 1901. The purpose of PCA in the context of feature transformation is to derive a new reduced set of features from the original feature set while taking into account the level of importance of each features. It is common practice to arrange the new feature set according to the decreasing measure of importance (represented by the variance).

As we mentioned earlier, the new feature set created by PCA is a linear combination of the original features, with the constraint that the new features have to be orthogonal to one another. Geometrically, it can be thought as a rotation of the original axes in the feature space to a new set of orthogonal axes that are ordered in terms of the amount of variation of the original features.

Although, it may be almost always possible to find a reduced representation of the original feature set, it is often difficult to assign meaningful interpretation on the new feature set. PCA is also a feature directed technique as it does not take into account of prior information such as the class label of the data. Thus it is essentially an unsupervised technique.

Given $f_1, \ldots f_p$ as the original set of features, and let $\xi_1, \ldots, \xi_{p'}$ be the new linear combination of these features:

$$\xi_i = \sum_{j=1}^{p} l_{ij} f_j \tag{5.9}$$

where l_{ij} is called the basis vector, and in matrix form, as we mention earlier in (5.8):

$$\xi = \mathbb{L}^T \mathbf{F} \tag{5.10}$$

Consider now the i^{th} new feature, we choose $\mathbf{a}_i = (a_{i1}, a_{i2} \dots a_{ip})^T$ to maximise the variance of f_i', subject to the constraint that $a_i^T a_i = 1$. The variance, i.e. the second central moment:

$$\begin{aligned} var(\xi_i) &= E\{\xi_i^2\} - E\{\xi_i\}^2 \\ &= E\{\mathbf{a}_i^T \mathbf{f}\mathbf{f}^T \mathbf{a}_i\} - E\{\mathbf{a}_i^T \mathbf{f}\} E\{\mathbf{f}^T \mathbf{a}_i\} \\ &= \mathbf{a}_i^T (E\{\mathbf{f}\mathbf{f}^T\} - E\{\mathbf{f}\} E\{\mathbf{f}^T\}) \mathbf{a}_i \\ &= \mathbf{a}_i^T \mathbf{C}_{\mathbf{ff}} \mathbf{a}_i \end{aligned} \tag{5.11}$$

To find the stationary value of (5.12), with the constraint $a_i^T a_i = 1$, is the same as to find the unconditional stationary value of the following:

$$f(\mathbf{a}_i) = \mathbf{a}_i^T \mathbf{C}_{\mathbf{ff}} \mathbf{a}_i - v \mathbf{a}_i^T \mathbf{a}_i \tag{5.12}$$

Hence, differentiating with respect to a_i and equating to zero gives:

$$\mathbf{C}_{\mathbf{ff}} \mathbf{a}_i - v \mathbf{a}_i = 0 \tag{5.13}$$

A trivial solution would be $\mathbf{a}_i = 0$, but this will then be meaningless. Therefore the nontrivial solutions of $\mathbf{a}_i$ have to be an eigenvector of $\mathbf{C}_{\mathbf{ff}}$ and v is the corresponding eigenvalues. Note that $\mathbf{C}_{\mathbf{ff}}$ has the rank of p, therefore we have p eigenvalues, $\lambda_1 \dots \lambda_p$, and we can rank them in descending order, without loss of generality:

$$\lambda_1 > \lambda_2 > \lambda_3 > \dots > \lambda_p$$

The i^{th} principal basis vector is then chosen as the eigenvector with the i^{th} largest eigenvalue λ_i since our objective is to maximise variance of ξ and by (5.12):

$$\mathbf{a}_i^T \mathbf{C}_{\mathbf{ff}} \mathbf{a}_i = \lambda_i \mathbf{a}_i^T \mathbf{a}_i \quad = \quad \lambda_i \tag{5.14}$$

By choosing the p' eigenvalues, the whitened version PCA transform has the closed form solution of:

$$\xi = \Lambda^{-1/2} \, \Sigma^T \mathbf{f} \tag{5.15}$$

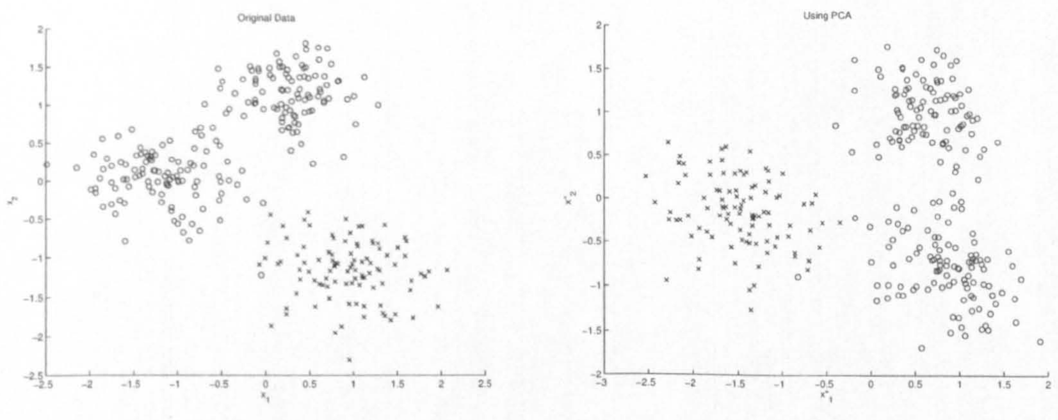
where Λ is diagonal matrix with it $i^t h$ value equal to λ_i in descending and Σ is a matrix with it's i^{th} column being the corresponding eigenvector.

More detailed discussions of PCA can also be found in [82].

Simple PCA Demonstration

In Figure 5.5, we show a simple demonstration where the application of PCA is particularly useful. Figure 5.5(a) shows three two-dimensional Gaussian clusters of data but with two data clusters belonging to the same class, denoted by 'o'. The third cluster is marked with 'x'. The first two clusters are of mean of '0' and '3' respectively. However, they are biased with a vector, $\mathbf{b}_{c_1} = [-3.0, 0.0]^T$. The third cluster has the mean of '0' but was biased with $\mathbf{b}_{c_2} = [1.5, -3.0]^T$. All clusters are of unit variance. From the figure, it is obvious that a non-linear separation boundary is required to separate the two classes accurately.

In Figure 5.5(b), the data is redrawn on the principal component axes, where



(a) Original Data (b) PCA transformed Data

Figure 5.5: A simple demonstration where PCA can be useful

it is obvious that by choosing the axes with maximum variance, a linear separation line can be readily drawn with the minimum mean squared error criterion on the first principal axes, x'_1. Instead of using both axes, only one principal axis is needed to do the classification (dimension reduction in the input features).

However, we note that the direction of maximum variance is not always the best direction.

PCA via Neural Network

Figure 5.6 shows an example of bottleneck multilayer perceptrons with linear activation function. Such a network can be used to create PCA basis vectors in an auto-associative mode. The number of output neurons is set equal to the number of input nodes and the number of hidden layer neurons is set to number of dimensions that one wishes to reduce to. This network can then be randomly initialised and trained with standard BP algorithm. The target vectors are set equal to the input vector. Upon convergence, The output layer can be pruned and the output of the hidden layer neurons will be the principal components of the input [83]. In other words, the weights of the hidden layer neurons act as the

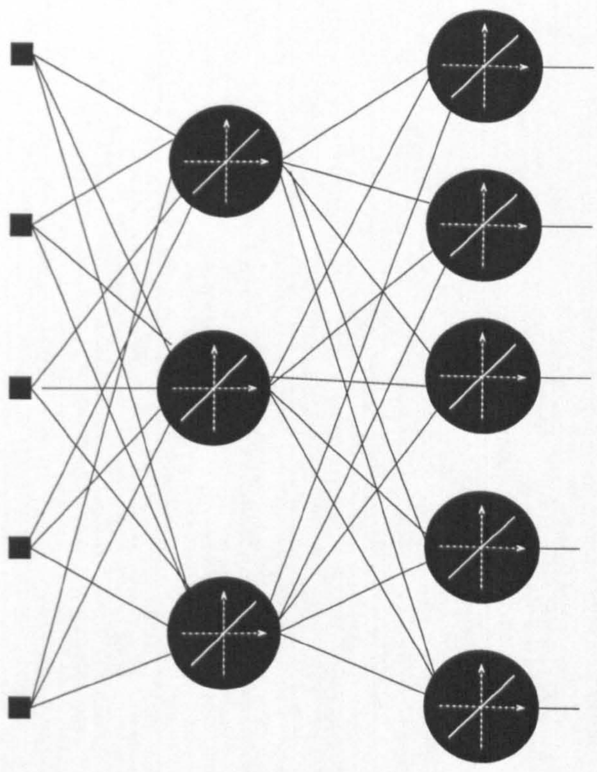


Figure 5.6: A linear bottleneck MLP network

principal basis vectors. Note that the activation function of the hidden layer has to be linear, otherwise the output would not be the principal components.

A non-linear version can also be easily implemented using a similar idea ([84]). To add in non-linearity, the network is extended to a five-layer network (Figure 5.7 with the addition of 2^{nd} and 4^{th} layer being non-linear activation neurons. The training of non-linear PCA network are the same as for linear PCA networks. At the end of training, the final layer is pruned off as in the linear case.

In this work, the experimental results were achieved by using the eigenvalue decomposition (EVD) approach; these neural networks (linear and non-linear) based PCA methods are included for completeness and interest of the readers. The investigation of the superiority of different PCA methods is beyond the context of this work. Interested readers can find out more in [85].

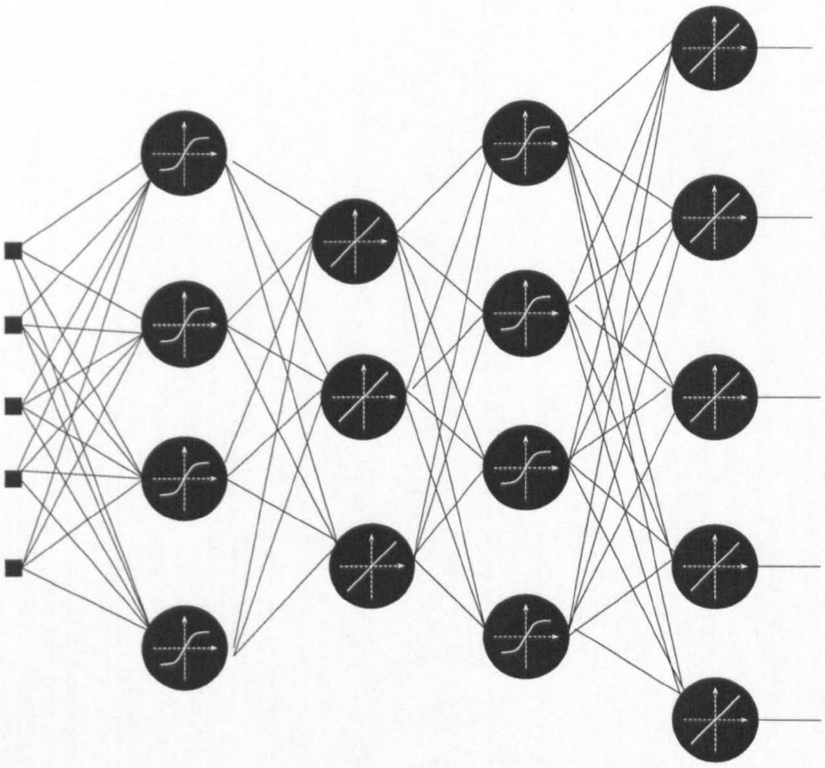


Figure 5.7: A non-linear bottleneck MLP network

5.3.2 Independent Component Analysis

Imagine the scene of a cocktail party, where multiple people are speaking simultaneously and yet the human ear have the amazing capability of recognising the individual sources. However, a recording of the party would contain a mixed up copy of different sources. The question here is, if multiple copies of the recording are obtained from multiple microphones, can one reconstruct the original sources? The answer to that is independent component analysis (ICA)

The name ICA was first coined by Comon in 1994 [86]. Nevertheless, ICA was originated form the work in solving blind source separation (BSS) problems (e.g. the cocktail party problem mentioned above) and the first paper was published by Herault and Jutten in 1991 [87]. A maximum likelihood solution was proposed in [88] and recently Hyvärinen et. al. proposed a FastICA algorithm that has been proven robust and useful in feature extraction of stereo images [89]. Other applications of ICA include extraction of fetus ECG diagram, artifacts cancellation of EEG, speaker separation, etc.

The application of ICA to feature transformation gets its motivation from results in neuroscience that suggest similar principle of redundancy reduction explains some respects of the early processing of sensory data by the brain [90, 91].

In ICA, we assumed that the observed data (the original feature set) are a linear combination of independent components (ICs). In statistical terms, these ICs are called latent sources, because they cannot be observed in the data. Considering only the case of linear mixtures, ICA adopts the following model:

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{5.16}$$

where $\mathbf{x}$ is the observed data, $\mathbf{s}$ are the original ICs and $\mathbf{A}$ is the unknown (assumed square) mixing matrix.

Estimating the ICs means that we are estimating a demixing matrix, denoted by $\mathbf{W}$:

$$\begin{aligned} \hat{\mathbf{s}} &= \mathbf{W}\mathbf{x} & (5.17) \\ &= \mathbf{W}\mathbf{A}\mathbf{s} \\ \therefore \mathbf{W} &= \mathbf{A}^{-1} & (5.18) \end{aligned}$$

By (5.18) we have implied the assumption that the mixing matrix is invertible, and estimating $\mathbf{W}$ is to find the inverse of $\mathbf{A}$. Unfortunately, $\mathbf{A}$ is unknown here.

However, by taking the following further assumptions, the problem can be simplified:

1. The ICs are assumed to be statistically independent. Mathematically this

means that:

$$P(s_1, s_2, \ldots, s_N) = \prod_{n=1}^{N} P(s_n) \qquad (5.19)$$

2. The probability distributions of the ICs are assumed to be non-Gaussian. Gaussian data is usually disregarded and treated as noise in ICA.

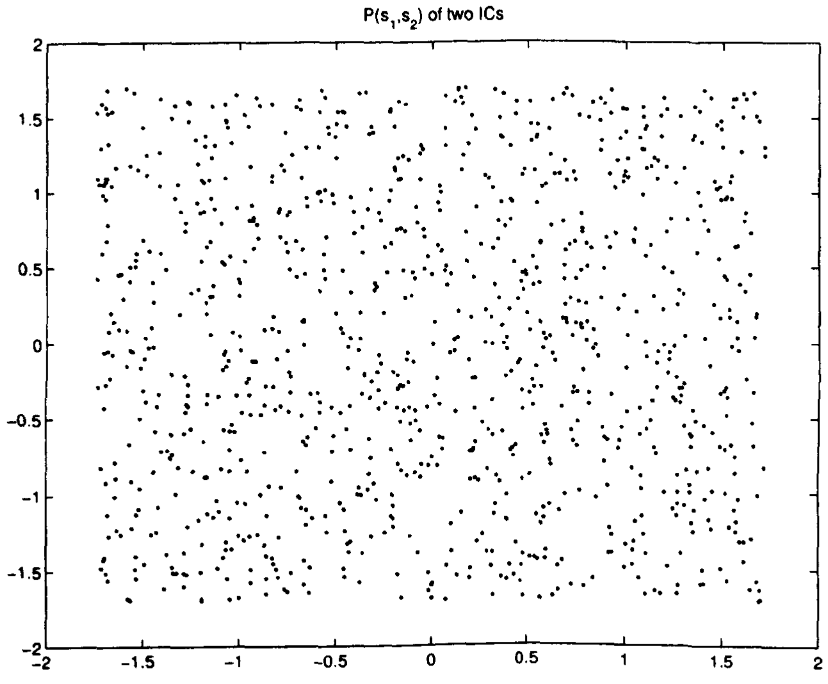


Figure 5.8: Joint distribution of two statistically independent sources

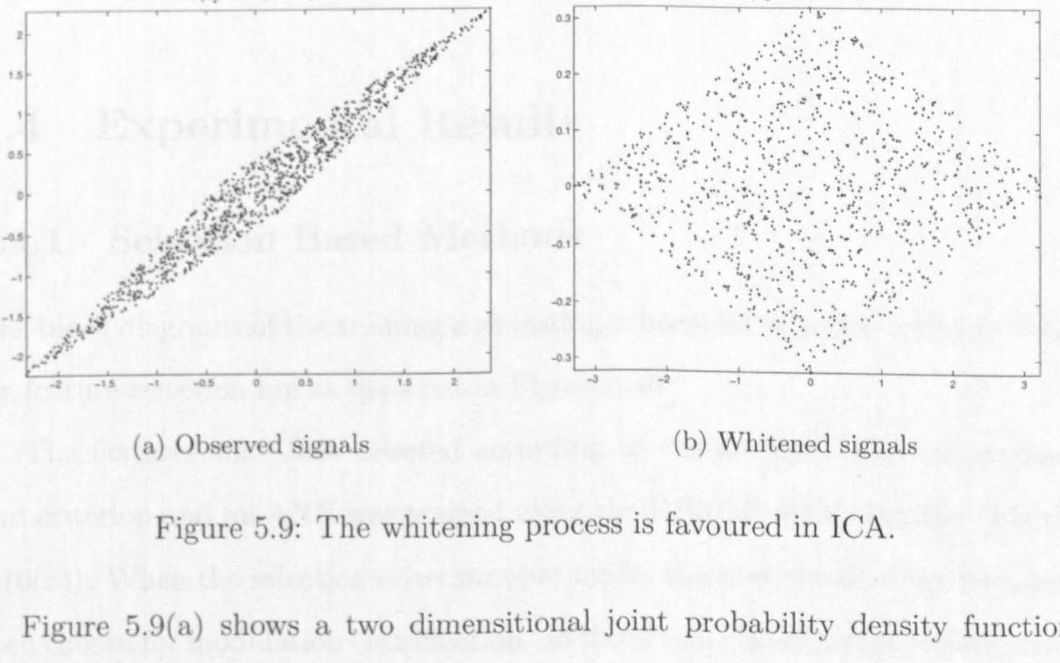
However, the output of ICA also possesses two main ambiguities:

1. The variances (energies) of the ICs cannot be determined. This means that the successfully recovered signal will be a scaled version of the original sources.

2. The order of the ICs cannot be determined, i.e. the original order of the sources can not be preserved. This has implication in some applications.

In many ways, ICA goes to PCA with whitening, and it is stricter than PCA as it imposes statistical independence for the sources while PCA only require uncorrelatedness. For PCA with whitening, the constraint is stricter but it is not enough for ICA. However, whitening is desired as a pre-process for ICA as it transforms the mixing matrix $\mathbf{A}$ to a new matrix $\tilde{\mathbf{A}}$ which is orthogonal. An orthogonal matrix contains $n(n-1)/2$ degree of freedom. For example, for a two source case, an orthogonal transformation is solely determined by an angle parameter. The mixtures is said to be whitened if:

$$E\{\mathbf{y}\mathbf{y}^T\} = \mathbf{I} \tag{5.20}$$



(a) Observed signals (b) Whitened signals

Figure 5.9: The whitening process is favoured in ICA.

Figure 5.9(a) shows a two dimensitional joint probability density function (scatter diagram) of two received signals while Figure 5.9(b) shows the same signals after PCA with whitening. Comparing Figure 5.9(b) with Figure 5.8, the whitened copy of the sources is a scaled copy of the original sources and offset with an angle. Finding the demixing matrix is now simplified to estimating the unknown angle.

In this part of the work, the FastICA package was used in the computer experiment. This package is freely distributed and downloadable via the package homepage on the Internet [1].

A summary of the FastICA algorithm for a single source can be found in Table A.1. The algorithm seeks to maximise the non-Gaussianity of the data through maximise the negentropy. It is a fast fixed point algorithm that approximate the negentropy using some non-linear functions, e.g. $g(x) = tanh(x)$. For the $m > 1$ sources case, an orthogonalisation stage is added to the algorithm. One way of doing so is using Gram-Schmidt method to estimate the sources one after another. Whenever p ICs have been estimated, the projection $(\mathbf{w}_{p+1}^T \mathbf{w}_i)\mathbf{w}_j$, for $j = 1, \ldots, p$ is subtracted from $\mathbf{w}_{p+1}$. The orthogonalisation stage prevents the vectors $\mathbf{w}_{1,\ldots,p}$ from having the same maximum. The algorithm is presented in Table A.2.

5.4 Experimental Results

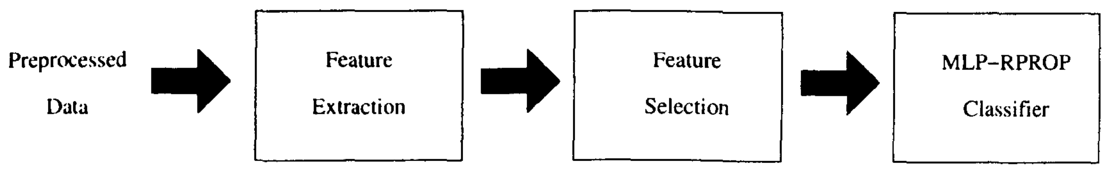
5.4.1 Selection Based Methods

The block diagrams of the training and testing process for selection based method for feature selection are as depicted in Figure 5.10:

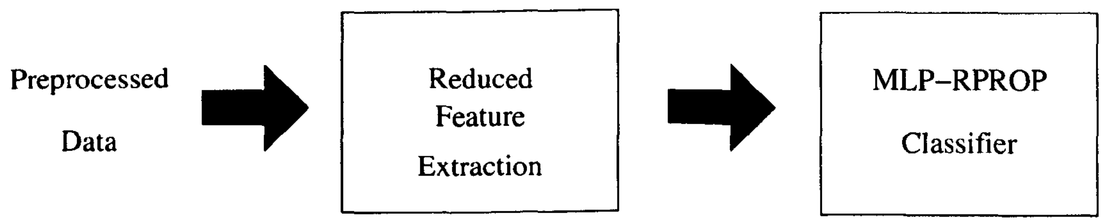
The features were first selected according to the assigned selection method and criterion and an ANN was trained using the RPROP MLP classifier (Figure 5.10(a)). When the selection criterion were made, the best classifier network was then chosen for modulation classification. In the actual classification process, only the required features were extracted from raw received data (Figure 5.10(b)).

In this section two variants of selection methods using GA were investigated. The first one is the binary coding method where the GA is allowed to search through 17 dimensions of feature space, and the latter being a fixed length list coding method where one has the freedom to choose the best pre-defined N'

[1]FastICA Homepage: `http://www.cis.hut.fi/projects/ica/fastica`



(a) During feature selection process



(b) After feature selection

Figure 5.10: Block diagrams for feature selection based MLP-RPROP Classifier

dimension.

Binary Coding

For binary coding, 20 binary genome strings were created randomly with uniform probability distribution during initialisation. This implies that each bit in the string is independent of other bits. One-point crossover and single bit mutation were used as reproduction operators. The GA was run until it reached convergence. Each feature subset selected was allowed to train for a maximum of 250 epochs unless it was stopped by the validation process prior to the maximum epoch being reached.

Table 5.4 shows the best genome strings selected in different SNR environments and their performance. These are then compared with the performance of the MLP classifier trained using RPROP.

Generally, performance from reduced feature sets selected in different SNRs show similar to the performance achieved by the RPROP MLP classifier. Moreover it should be noted that, the GA achieved this performance using only 10 to 12 features typically, with the exception of SNR = -5 dB, which required 14 features.

SNR	Binary String	Features Chosen	Perf. with GA	RPROP Perf.
-5 dB	11011111111111100	14	94.37%	94.79%
0 dB	11110100011101111	12	99.15%	99.07%
5 dB	11110100010111101	11	99.97%	99.53%
10 dB	11110100011101111	12	99.97%	99.95%
20 dB	11011000001110111	10	99.99%	99.97%

Table 5.4: Performance of binary string GA RPROP MLP with 10 hidden layer neurons with dynamic feature selection

Fixed Length List Coding

The experimental setup remained similar to the previous experimental setup. However, the following results were obtained through running the experiments in compiled environment using a software suite written entirely in C++ [2]. The advantage being compiled language offer much efficiency and consumed less CPU time for computer experiments than interpreted language such as MATLAB. As the lists of experimental were a lot more elaborate than the binary string GA feature selection, using compiled language was a natural choice.

With list genome encoding method, one has to specify the length of the genome string, which represents the number of features in the reduced feature set. As seen from the previous section, without any compromise in performance, the feature set can typically be reduced down to twelve features. Therefore in this section, experiments were carried out to investigate the possibility of even smaller datasets and the compromise in performance that might be observed.

At each SNR, a GA was used to select a set of designated number of features.

[2]The author would like to express his thanks to Mr. A. D. Parkins for sharing his GAFFS package [92].

Here, the total number of features to be selected ranges from one single feature to eight features. The lists of feature selected for -5 dB, 0 dB, 5 dB, 10 dB, and 20 dB are shown in Tables 5.5, 5.6, 5.7, 5.8 and 5.9 respectively. A summary of the classification performance at each SNRs is then found Table 5.10.

Required No of Features	**Features Selected**
1	5
2	1 6
3	1 6 14
4	1 6 8 12
5	1 6 9 11 14
6	1 8 9 11 14 16
7	1 6 8 9 10 11 14
8	1 5 8 10 11 12 14 16

Table 5.5: Features selected by GA at -5 dB SNR

Required No of Features	**Features Selected**
1	5
2	5 11
3	1 6 14
4	1 2 6 14
5	1 6 9 11 14
6	1 4 6 7 8 14
7	1 2 4 6 8 9 14
8	1 5 6 7 9 11 12 14

Table 5.6: Features selected by GA at 0 dB SNR

Each GA was stopped when the convergence criteria were reached. Figure 5.11 to Figure 5.13 show some of the GA process plots for 0 dB, 10 dB and 20 dB SNR respectively. It is clear that all GAs achieved convergence generally and 25 generations were typically adequate. Moreever, from these plots, it is shown that the processes with 10 dB and 20 dB SNR show quicker convergence than those with 0 dB SNR.

From Table 5.10, it was shown that by using only 3 features one can achieve

Required No of Features	Features Selected
1	5
2	1 6
3	5 6 14
4	1 3 6 14
5	5 6 7 11 14
6	3 5 6 7 11 14
7	1 2 3 4 7 12 14
8	1 3 4 5 7 11 14 15

Table 5.7: Features selected by GA at 5 dB SNR

Required No of Features	Features Selected
1	1
2	1 6
3	5 6 14
4	1 3 6 14
5	1 3 7 12 14
6	3 6 11 12 13 14
7	0 1 3 6 7 12 14
8	0 1 3 5 7 10 11 14

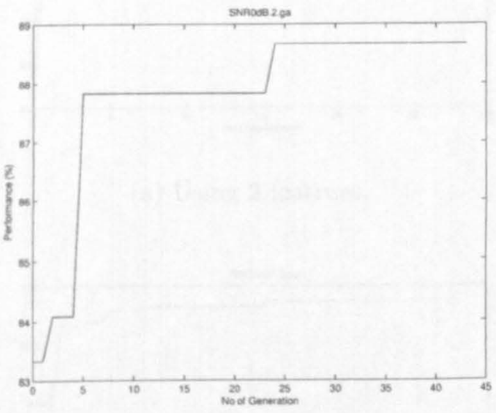
Table 5.8: Features selected by GA at 10 dB SNR

Required No of Features	Features Selected
1	1
2	1 6
3	1 3 12
4	1 3 5 14
5	1 3 5 7 14
6	1 3 7 12 14 15
7	0 1 3 5 6 7 14
8	0 1 2 3 5 6 14 15

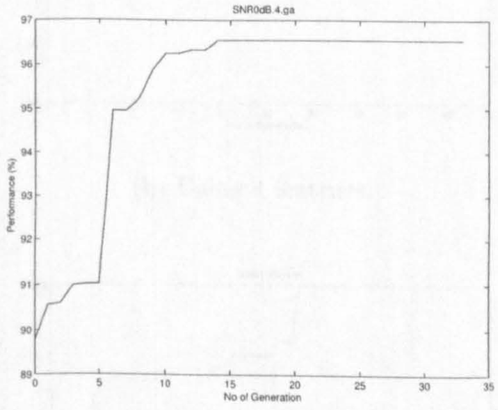
Table 5.9: Features selected by GA at 20 dB SNR

Number of	SNR				
required features	**-5 dB**	**0 dB**	**5 dB**	**10 dB**	**20 dB**
1	66.68	65.24	67.95	64.06	68.23
2	83.97	88.67	90.92	91.04	93.82
3	87.05	96.79	98.85	99.32	98.57
4	88.15	96.56	98.61	99.73	99.95
5	90.13	97.59	99.24	99.80	99.98
6	90.92	96.88	99.17	99.87	99.98
7	91.33	97.02	99.12	99.88	99.99
8	90.79	98.39	99.25	99.90	99.98

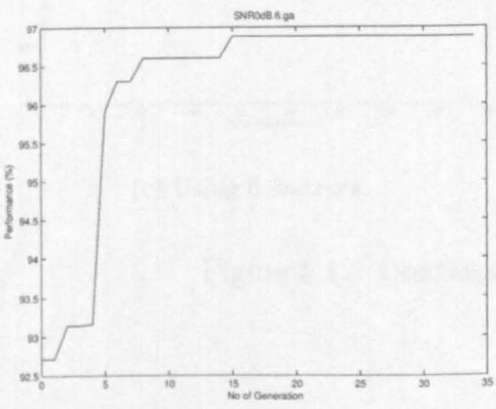
Table 5.10: Performance of the GA aided MLP RPROP classifier



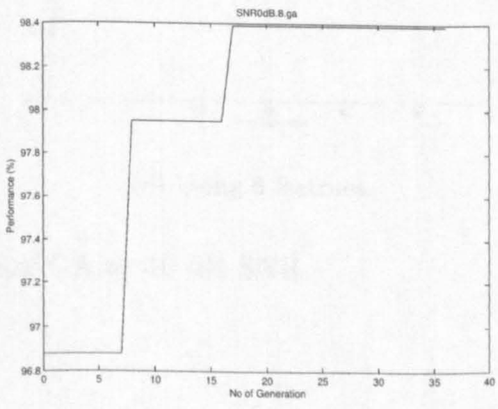
(a) Using 2 featrues.



(b) Using 4 featrues.

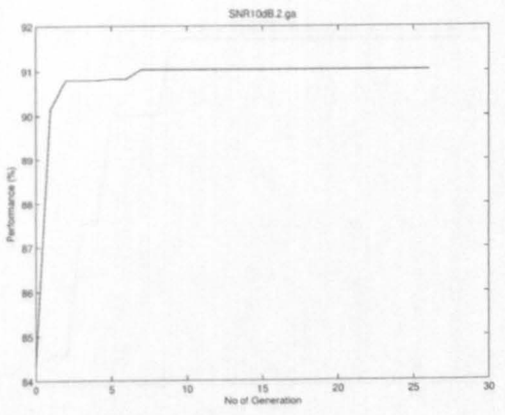


(c) Using 6 featrues.

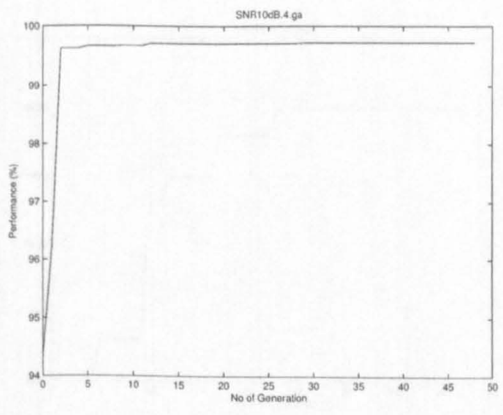


(d) Using 8 featrues.

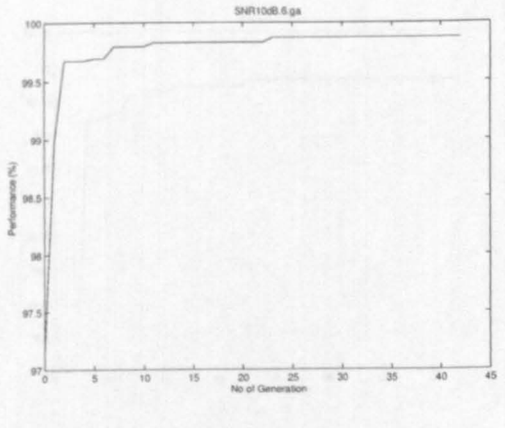
Figure 5.11: Convergence of GA at 0 dB SNR



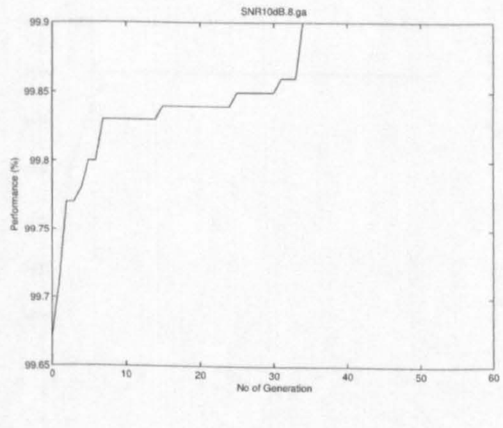
(a) Using 2 featrues.



(b) Using 4 featrues.

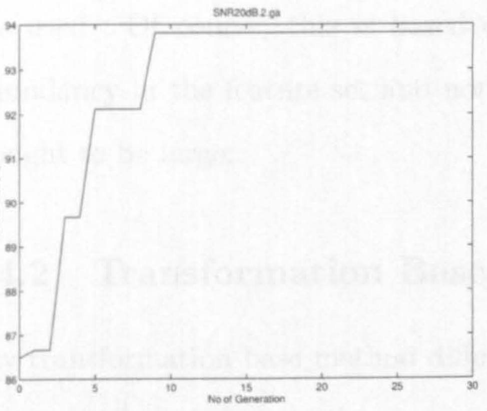


(c) Using 6 featrues.

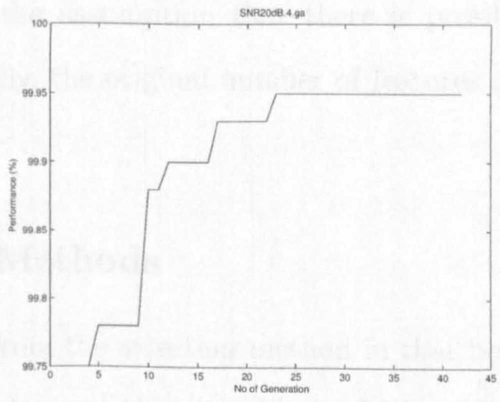


(d) Using 8 featrues.

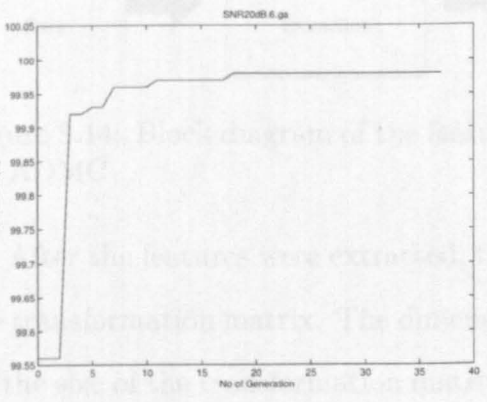
Figure 5.12: Convergence of GA at 10 dB SNR



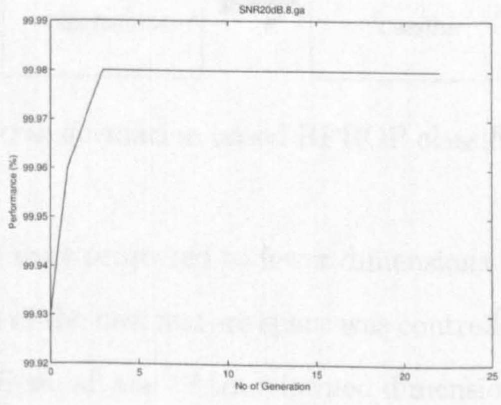
(a) Using 2 featrues.



(b) Using 4 featrues.



(c) Using 6 featrues.



(d) Using 8 featrues.

Figure 5.13: Convergence of GA at 20 dB SNR

$> 95\%$ in classification accuracy for SNRs larger than 0 dB. By increasing the number features, better performance were achieved in the low SNRs. These results demonstrated the advantage and practicality of the proposed GA feature selection algorithm. By using a fraction of the original feature set, one can achieve good classification results similar to the results achieved when a full feature set was used. Of course, this is based on the assumption that there is possible redundancy in the feature set and normally, the original number of features are thought to be large.

5.4.2 Transformation Based Methods

Any transformation base method differs from the selection method in that both training and testing share the same procedures as shown in Figure 5.14:

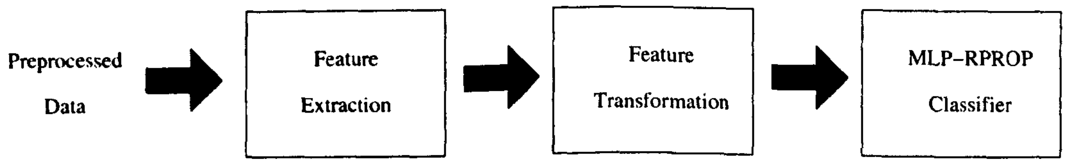


Figure 5.14: Block diagram of the feature transformation based RPROP classifier for ADMC

After the features were extracted, they were projected to fewer dimensions by the transformation matrix. The dimension of the new feature space was controlled by the size of the transformation matrix. Here, all the 17 transformed dimensions were investigated because it was quicker to calculate than the GA based feature selection method.

For each set of experiments, the RPROP-MLP was allowed to train for a maximum of 250 epochs as with the GA based feature selection. The architecture for the RPROP-MLP remained unchanged from previous section, i.e. a three layer network, with 10 hidden-layer neurons. The validation process was also used for experiments. The transformation matrix was first found for each individual

SNR and the N most influential (components with most significant eigenvalues) were chosen for dimensionality reduction. All features in the datasets were pre-normalised before PCA and whitening.

To estimate the number of reduced components needed, the cumulative sum of the eigenvalues of the pre-normalised training dataset for each SNR were plotted on Figure 5.15. From the figure, it was shown that for SNR above 10 dB, the eigenvalues were stable and for most SNRs except 0 dB and below, eight components were enough to claim more than 90% contribution towards to cumulative sum of eigenvlaues. In any case, 11 components (all datasets converged to the same percentage at this level) claimed over 97.5% of the cumulative sum of the eigenvalues. These useful information could facilitate the analyses of the results shown in Table 5.11.

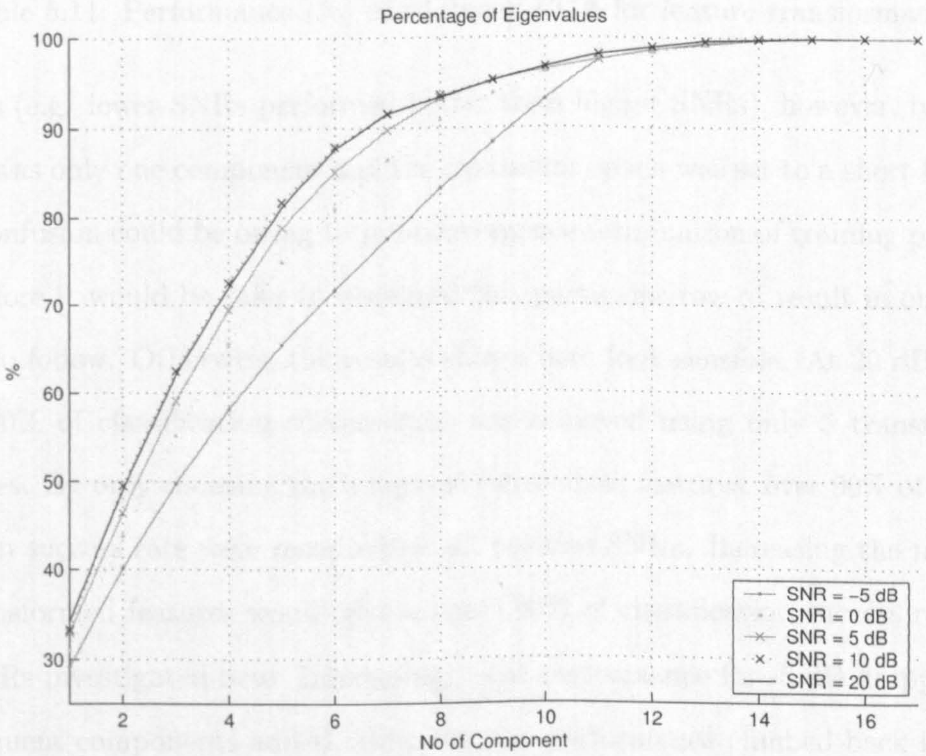


Figure 5.15: Percentage of eigenvalues against the number of selected components

From Table 5.11, the results recorded for single component showed some mixed

No. of	SNR in dB				
Features	-5	0	5	10	20
1	69.8	57.7	57.2	75.1	70.8
2	72.1	76.8	83.7	79.7	89.0
3	76.2	81.7	87.0	83.2	92.3
4	79.3	84.2	87.0	84.7	93.8
5	81.9	87.8	88.6	87.3	95.9
6	85.0	91.6	92.6	96.9	97.1
7	83.4	94.9	97.3	98.9	99.7
8	84.3	95.4	97.5	99.5	99.9
9	90.6	95.9	98.3	99.7	99.9
10	85.5	96.2	98.5	99.7	99.9
11	85.5	95.9	98.7	99.7	99.9
12	88.3	96.2	98.8	99.9	99.9
13	88.0	96.5	99.2	99.7	99.9
14	92.4	97.6	99.3	99.9	99.9
15	92.3	98.7	99.3	99.9	99.9
16	93.0	99.1	99.6	99.9	99.9
17	93.8	99.2	99.7	99.9	99.9

Table 5.11: Performance (%) of whitened PCA for feature transformation

results (e.g. lower SNRs performed better than higher SNRs), however, because there was only one component and the maximum epoch was set to a short length, this confusion could be owing to pre-convergence termination of training process. Therefore it would be safer to disregard this particular row of result in our analyses to follow. Otherwise, the results shown here look sensible. At 20 dB SNR, over 90% of classification success rate was achieved using only 3 transformed features. By only choosing the 6 highest eigenvalues features, over 90% of classification success rate were recorded for all positive SNRs. Increasing the number of transformed features would give us over 90% of classification success rate for all SNRs investigated here. Interestingly, the performance for -5 dB dropped for subsequent components added. However, the performance climbed back to 92% at 14 components, and stabilised for the rest of the experiments.

For ICA, the procedure was largely the same as for the PCA. The number of input dimension were first determined using N components with largest eigenvalues as for PCA. Subsequently, the reduced feature set (whitened) were orthogonalised using deflation method as described in Table A.2. The RPROP-MLP used for this investigation was identical in structure with the one used for PCA investigation. The results were shown in Table 5.12

No. of	SNR in dB				
Features	-5	0	5	10	20
1	69.69	58.60	57.15	75.13	77.98
2	69.14	79.69	83.73	77.45	89.45
3	78.37	82.45	86.84	82.75	91.45
4	82.72	85.02	86.62	92.43	93.91
5	78.28	88.70	88.74	92.87	95.00
6	83.76	91.69	91.68	97.79	97.95
7	90.50	95.71	98.50	98.84	99.80
8	92.83	95.89	97.66	99.64	99.87
9	88.85	98.71	99.63	99.62	99.84
10	93.77	95.23	98.79	99.77	99.93
11	87.28	96.32	99.05	99.88	99.92
12	87.57	96.52	98.82	99.83	99.90
13	93.04	98.83	99.47	99.20	99.88
14	93.08	97.13	99.38	99.79	99.93
15	94.01	97.73	99.69	99.91	99.94
16	94.29	99.07	99.76	99.92	99.87
17	94.53	99.20	99.78	99.91	99.90

Table 5.12: Performance (%) of ICA (with pre-normalisation of features) for feature transformation

From the table, the performance recorded using ICA transformation was largely similar to those shown by using whitened PCA transform. In other words, ICA had little to offer in improving the performance of the datasets used in this investigation. The performance of higher dimension at this SNR were nearer to the original RPROP performance value (Table 5.4).

Figures 5.16 and 5.17 showed the two transformed features with largest eigenvalues after whitening and after ICA respectively at SNR of 20 dB. The horizontal

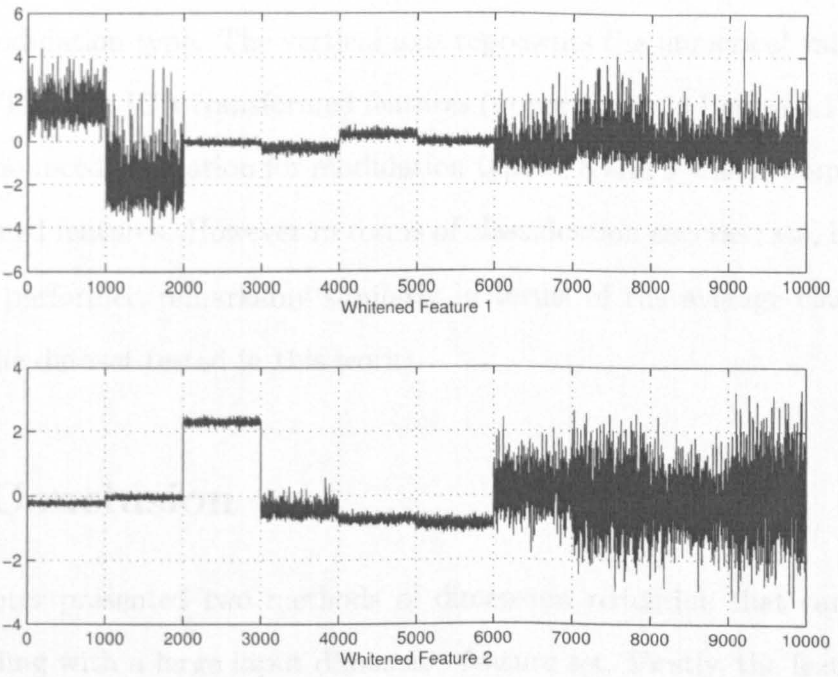


Figure 5.16: The top two whitened PCA transformed features

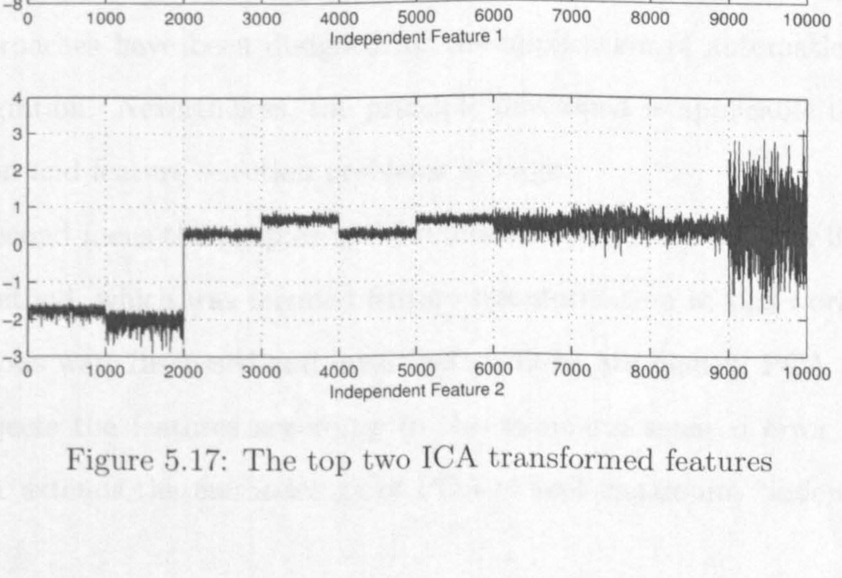


Figure 5.17: The top two ICA transformed features

axis represents the number of samples, and each 1,000 samples belongs to a particular modulation type. The vertical axis represents the numerical values of the features. The first ICA transformed features (upper figure in Figure 5.17) showed more pronounced separation for modulation type 7, 8 and 9 when compared with the whitened features. However in terms of classification success rate, both PCA and ICA performed remarkably similarly in terms of the average classification rate for the dataset tested in this work.

5.5 Conclusion

This chapter presented two methods of dimension reduction that can be used when dealing with a large input dimension feature set. Firstly, the feature selection based method were demonstrated through the introduction of GA in digital modulation classification. Two approaches of selection were investigated: the binary string genome method and the list string genome method. The first approach enables the search through the entire feature space and find the best combination of features. The number of features selected are determined by the algorithm. The latter approach, however, gave more control to the user in that one has the choice of fixing the number of features required for the application. Both approaches have been designed for the application of automatic modulation recognition. Nevertheless, the principle developed is applicable to pattern recognition and feature selection problems at large.

The second focus the chapter is the dimension reduction through linear projection method, which was terrmed feature transformation in this work. Again, two methods were discussed and investigated; these are namely PCA and ICA. PCA projects the features according to the minimum squared error criterion, while ICA extends the methodology of PCA to seek maximum "independence" property among the features. Both method gave similar performance as shown in the results recorded from computer experiments.

In term of performance, the list string genome based GA showed better performance when compared with the feature transformation based method, particularly in lower dimension feature space ($N = 2$ or 3). However, the feature transformation based method offers quicker operational time. Nevertheless, the transformed feature space were difficult to interpret as they were not direct subset, but combinations, of the original feature space.

Chapter 6

Maximum Likelihood Classification of Digital Modulations

ANN based classifiers exhibit high degree of performance (Chapter 4 and 5) in correctly recognising a wide variety of modulations across a range of SNRs. However, these appealing classification performances depend on the availability of training examples. The number of training examples required is related to the number of features in the input feature set to the ANN classifier. Although ANNs are well known for their robustness, the re-training requirement at designated SNRs are troublesome and time consuming.

There are situations where training examples are not available or the number of available training examples are not sufficient for the training of an ANN classifier. The parameters of the operating environment (e.g. SNR value) might also change rapidly, hence re-training of classifiers becomes impractical and something to avoid. These are realistic factors where unsupervised algorithms become appealing. By unsupervised, it is meant that the decision boundaries of the classifier are drawn without using the knowledge of existing examples. Instead, other prior knowledge of the modulations are exploited. In this work, we investigate a family of classifiers that exploit the probabilistic relationship between the received samples and source constellations. These are namely, the Maximum Likelihood (ML) modulation classifier, the Estimated Maximum Likelihood (EML) classifier, and a the Minimum Distance (MD) modulation classifier.

We begin by some discussions on The Law of Total Probability and Bayes Rules for estimation. Following which the ML modulation classifier (MC) is presented. The EML MC is a version of the ML where the unknown SNR is estimated through a proposed simple algorithm. The MD MC is a reduced version of the ML MCs proposed with the aim to reduce the complexity of ML MC. The performances of these classifiers were investigated first in coherent environment and then in non-coherent environment.

When classifying in non-coherent environment a novel method of utilising a well known EML closed form ICA/BSS estimator for phase mismatch removal. This removal process is essentially a blind process as we do not have the information of phase mismatch on arrival of the received signal.

The performance of the classifiers are evaluated through computer simulation in MATLAB environment, and selected results are reviewed at the end of this chapter.

6.1 The Law of Total Probability and Bayes Rules

The *law of total probability* states if an event A can exist in m states, and each of the m states occurs exclusively, then the the probability of event A happening is equal to the sum of all the probabilities of the occurrence of the m states. For example, if a random variable y can take m different values of y when x takes the values of $x_1, x_2, \ldots, x_m$, then $P(y)$ is the sum of the joint probability $P(x, y)$ over all possible values of x. Mathematically, the law of total probability is stated as follows:

$$P(y) = \sum_{x \in \mathcal{X}} P(x, y) \tag{6.1}$$

When two variables are statistically dependent, the information of one of the two will help in estimating the other. This is called the conditional probability, $P(x \mid y)$ i.e. the probability of event x given event y is observed, and is stated mathematical as follows:

$$P(x \mid y) = \frac{P(x, y)}{P(y)} \tag{6.2}$$

and equally:

$$P(y \mid x) = \frac{P(x, y)}{P(x)} \tag{6.3}$$

Combining (6.1) with (6.2) and (6.3); rearranging them gives us the Bayes theorem:

$$P(x \mid y) = \frac{P(y \mid x)P(x)}{\sum_{x \in \mathcal{X}} P(y \mid x)P(x)} \tag{6.4}$$

and if x takes M states, for clarity we can write:

$$P(x_j \mid y) = \frac{P(y \mid x_j)P(x_j)}{\sum_{i=1}^{M} P(y \mid x_i)P(x_i)} \tag{6.5}$$

In pattern recognition, y often represent the observation and $x_1, x_2, \ldots x_M$ are the various pattern classes, (or as the causes in statistics) Bayes' theorem can be expressed in term of words as the following:

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}} \tag{6.6}$$

The term likelihood, $P(y \mid x_j)$ expresses how likely the observation is to occur given that the underlying class is actually x_j, it is easier to estimate because we would know what the pattern (cause) is. On the other hand, the posterior probability $P(x_j \mid y)$ is difficult to determine as often the observation can be originated from a several causes. Nevertheless, Bayes' theorem inverts this situation and makes it easy to estimate the posterior given that we know the likelihood and the prior probability of each class, $P(x_i)$. The denominator in (6.5) is a normalising factor to ensure that the posterior probabilities sum up to unity. Therefore it is the numerator which actually gives the shape of the function.

It is then logical, given an observation and a set of classes for classification, one would choose the observation-class pair which give the maximum posterior probability. This plain logic gives us the fundamental of a particular classifier family, known as the maximum *a posteriori* (MAP) classifier. However, in many cases, it is fair to assume that all classes in the system share the same prior probability, therefore the shape of function is now only depending on the likelihood term since the prior term is now constant. This then bring us to the maximum likelihood method for classification which chooses the observation-class pair that possesses the highest likelihood measure.

6.2 Coherent Maximum Likelihood Classifier

The received signal can be modeled ((3.1), Chapter 3) as the following:

$$y(n) = A \cdot \sum_{t=-\infty}^{\infty} x(\ell)h(nT - \ell T + \epsilon_T T) \cdot \exp(j2\pi f_o T_n + j\theta_n) + g(n) \qquad (6.7)$$

where all the variables are as defined previously in Chapter 3. In this section, we assume ideal working condition with only the presence of white Gaussian noise.

The communication channel had been adequately equalised and the residual channel effect, $h(\cdot)$ is negligible. Besides, other parameters such as symbol timing, T, carrier frequency, f_c, etc are known or have been successfully estimated. In some literature, e.g. [32, 34, 33], it is assumed that the signal amplitude and the noise variance are also known, i.e. one has knowledge of the SNR. This assumption can be relaxed as will be shown here through experimental results. One other important assumption is that the transmitted symbols are independently and identically distributed (i.i.d) process. The signal and noise are also not correlated.

It was shown in [32] that under these assumptions, the received signal sequence, after being processed by the quadrature receiver, is a sufficient statistic for modulation classification. The received sequence in ideal coherent environment can then be rewritten as:

$$y(n) = Ax(n) + g(n) \tag{6.8}$$

As noted before, each modulation is associated with one particular constellation which is a set complex points. Let us denote a group of c possible constellations by

$$I_l = \{x_{i1}, x_{i2}, x_{i3}, \dots, x_{iM_i}\}, \qquad l = 1, 2, \dots, c \tag{6.9}$$

where M_l is the total number of symbols in constellation, I_l.

It was mentioned briefly in Section 3.2 that classification of a set of constellation can be thought of as a hypotheses testing problem on the following c hypotheses:

$$H_l = \text{the underlying constellation in } I_l \qquad l = 1, 2, \dots, c \tag{6.10}$$

The maximum likelihood (ML) solution for classifying different modulation types is to choose the hypothesis that has the largest value of its likelihood function give a set of received data, $Y_N = \{y_k = [y_{Ik}, y_{Qk}]^T\}, \quad k = 1, 2, \ldots, N$:

$$\begin{aligned} H_l^* &= \arg\max_{H_l} L(H_l \mid Y_N) && (6.11) \\ &= \arg\max_{H_l} P(Y_N \mid H_l) \end{aligned}$$

6.2.1 The Likelihood Function

Using the law of total probability (Equation (6.1)), we have

$$P(y_k \mid H_l) = \sum_{i=1}^{M_l} P(x_{li} \mid I_l) P(y_k \mid x_{li}) \tag{6.12}$$

Under AWGN environment, the likelihood is as follows:

$$P(y_k \mid x_{li}) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2} \|y_k - x_{li}\|^2) \tag{6.13}$$

where σ^2 represents the variance of the Gaussian noise. With the i.i.d assumption, we can assume each constellation point has equal probability of being observed, therefore we can write:

$$P(y_k \mid H_l) = \sum_{i=1}^{M_l} \frac{1}{M_l} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2} \|y_k - x_{li}\|^2) \tag{6.14}$$

Taking into account of all received point, The likelihood function is then:

$$\begin{aligned} L(H_l \mid Y_N) &= \prod_{k=1}^{N} P(y_k \mid H_l) && (6.15) \\ &= \prod_{k=1}^{N} \sum_{i=1}^{M_l} \frac{1}{M_l} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2} \|y_k - x_{li}\|^2) && (6.16) \end{aligned}$$

It is often easier to take the natural logarithm of the likelihood function, and because the natural logarithm is a monotonic function, taking the maximum of the logarithm achieve the same result as taking the maximum of the likelihood function.

$$
\begin{aligned}
\ell(H_l \mid Y_N) &= \ln\{L(H_l \mid Y_N)\} \\
&= \ln\{\prod_{k=1}^{N}\sum_{i=1}^{M_l}\frac{1}{M_l}\frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}\|y_k - x_{li}\|^2)\} \\
&= \sum_{k=1}^{N}\ln\{\sum_{i=1}^{M_l}\frac{1}{M_l}\frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2\sigma^2}\|y_k - x_{li}\|^2)\}
\end{aligned}
\quad (6.17)
$$

In practice, the constant term can be dropped, and the log likelihood equation is reduced to:

$$
\ell(H_l \mid Y_N) = \sum_{k=1}^{N}\ln\{\sum_{i=1}^{M_l}\frac{1}{M_l}\exp(-\frac{1}{2\sigma^2}\|y_k - x_{li}\|^2)\} \quad (6.18)
$$

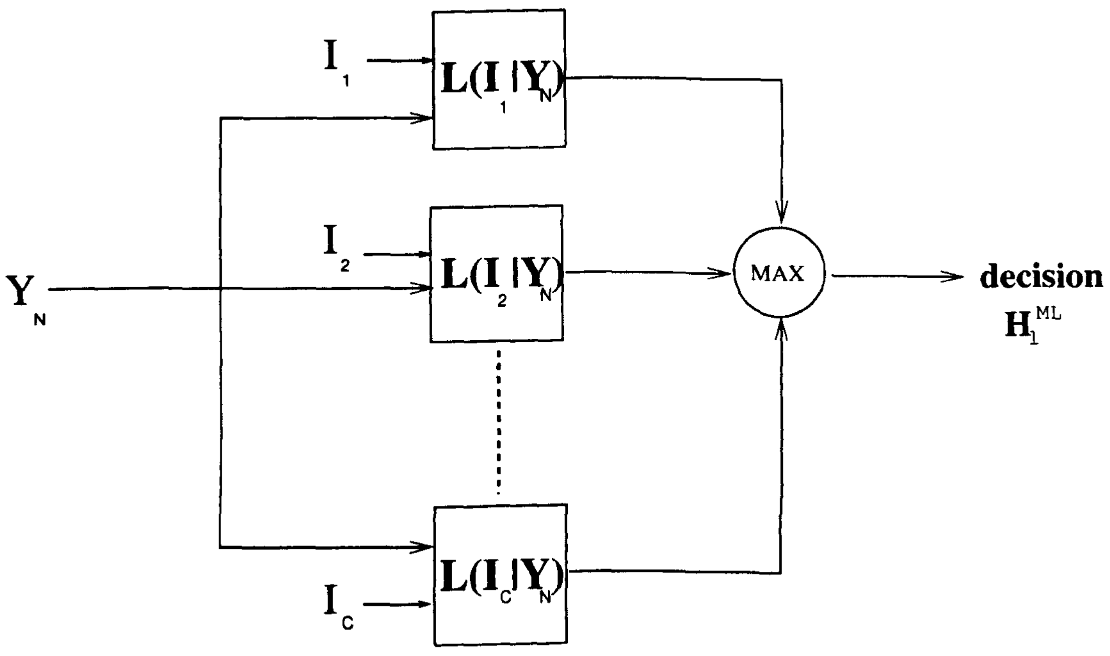


Figure 6.1: The maximum likelihood modulation classifier

Figure 6.1 shows a block diagram of the maximum likelihood classifier for modulation recognition. Upon availability of the output of the quadrature filters, the signal is then passed to a bank of likelihood estimators. Each estimator has a knowledge of a particular type of constellation diagram. The outputs of these (log)likelihood estimators are then processed by a maximum operator (competitive layer in ANN terminology). This method accommodates any modulation type with constellation diagram that can be expressed in its in-phase (I) and quadrature (Q) components. Although, the classifier is only optimum in the ideal coherent assumption, it nevertheless gives an upper bound for classification performance in comparison with the non-ideal scenario. In real life, the recovered constellation diagram is affected by frequency offset which results in a phase mismatch between the received data and the original constellation.

6.2.2 Variability of Likelihood Function caused by SNR

Both the likelihood function and its logarithm acquire the knowledge of the noise variance, σ^2. This is readily available if one has the knowledge on the working SNR. The parameter σ^2 scale the width of the curve of the probability density function (PDF). Three one dimensional Gaussian PDFs (i.e. the likelihood function in this case) are shown in Figure 6.2. A PDF with higher SNR (i.e. small variance) will have a sharper peak than one with lower SNR. The horizontal axis represents the distance of the received signal point and the constellation point. In any case, the likelihood function has a peak when the distance is the smallest. When SNR is infinitely high, the PDF reduced to a point source on the two dimensional PDF (the constellation diagram).

However, in many cases, the knowledge of SNR is not available directly. Therefore, the noise variance need to be estimated. A simple and approximate method of estimating the noise variance is presented in the following section:

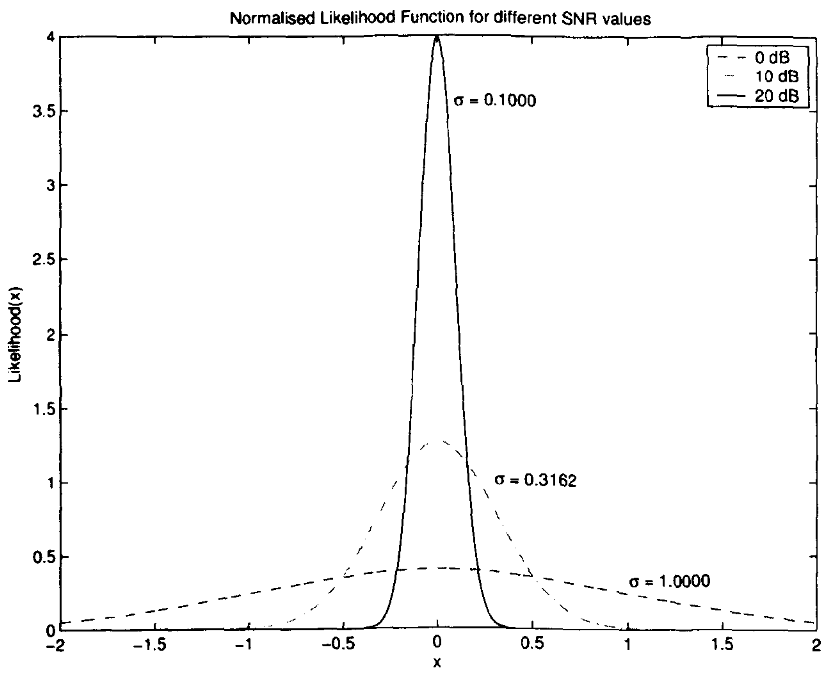


Figure 6.2: Likelihood function with various values of σ

Estimation of the Unknown σ Parameter

As with the formulation of the maximum likelihood classifier, we assumed that we are working in an ideal coherent AWGN channel. The noise variance can then be estimated through the mean squared distance between the received signal point and the original constellation point.

$$\hat{\sigma}_l^2 \approx \sum_{k=1}^{N} d(y_k, x_{li}^*) \tag{6.19}$$

where

$$x_{li}^* = \arg\min_{x_{li}} d(y_k, x_{li}) \qquad i = 1, 2, \ldots, M_l \tag{6.20}$$

and

$$d(m, n) = \|m - n\|^2 \tag{6.21}$$

By the definition, the source constellation point is unknown, therefore for a group of c modulation type, there are c estimated noise variances. Figure 6.3 shows the matching pair of signal and source constellation will have the smaller value of the estimated variance at high SNR region (SNR $\geq$ 10 dB). At these SNRs, the mismatch pairs always exhibit a higher value than the matching pair. Therefore, with this observation, our variance estimator logically chooses the minimum of $\hat{\sigma}_l^2$ for $l = 1, 2, \ldots, c$:

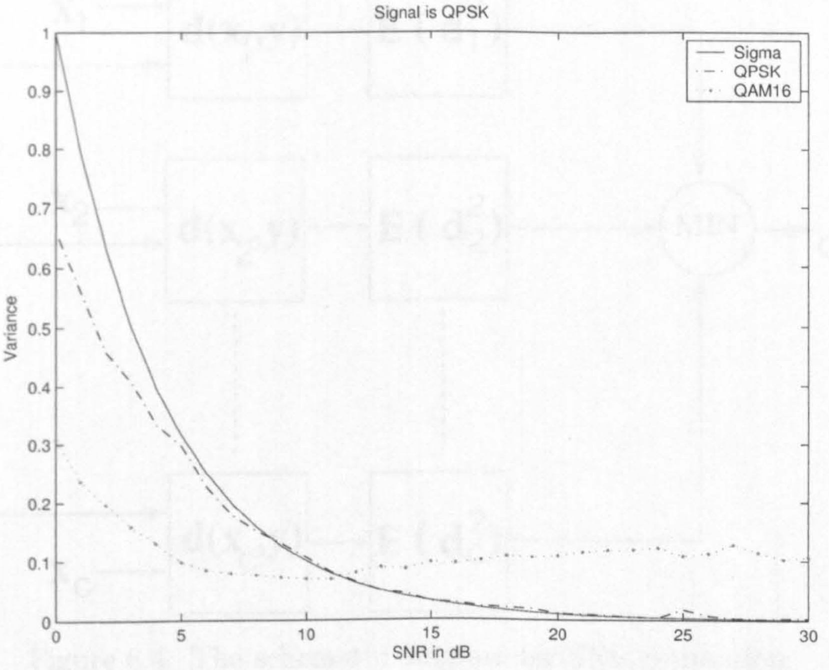
$$\hat{\sigma}_l^{*2} = \arg\min \hat{\sigma}_l^2 \quad l = 1, 2, \ldots, c \tag{6.22}$$

A block diagram of such an estimator is shown in Figure 6.4. For lower SNR, the variance estimator does not perform well. However, the maximum likelihood classifier shows a good degree of robustness against the estimation error, as will be shown later, as long as the σ^2 is common to all sub estimator modules within the classifier.

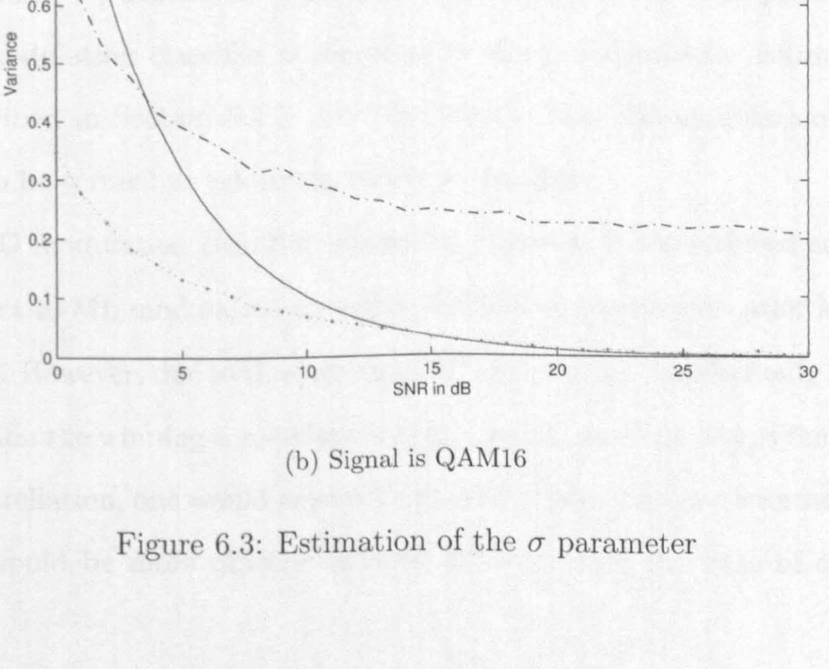
6.2.3 Minimum Distance Classifiers

ML modulation classifier provides an optimum performance for benchmarking in AWGN environment. However, it would be beneficial to reduce the complexity of the classifier as the speed is an important factor in modulation classification. From (6.18) and Figure 6.2, it was observed that the maximum of the likelihood functions occur when the distances between the received signal and the constellation point are at the minimum ($d = 0$). This observation suggests an alternative classifier based on the distances only.

Given the received signal, y_k and a set of c constellations, I_l where l =



(a) Signal is QPSK



(b) Signal is QAM16

Figure 6.3: Estimation of the σ parameter

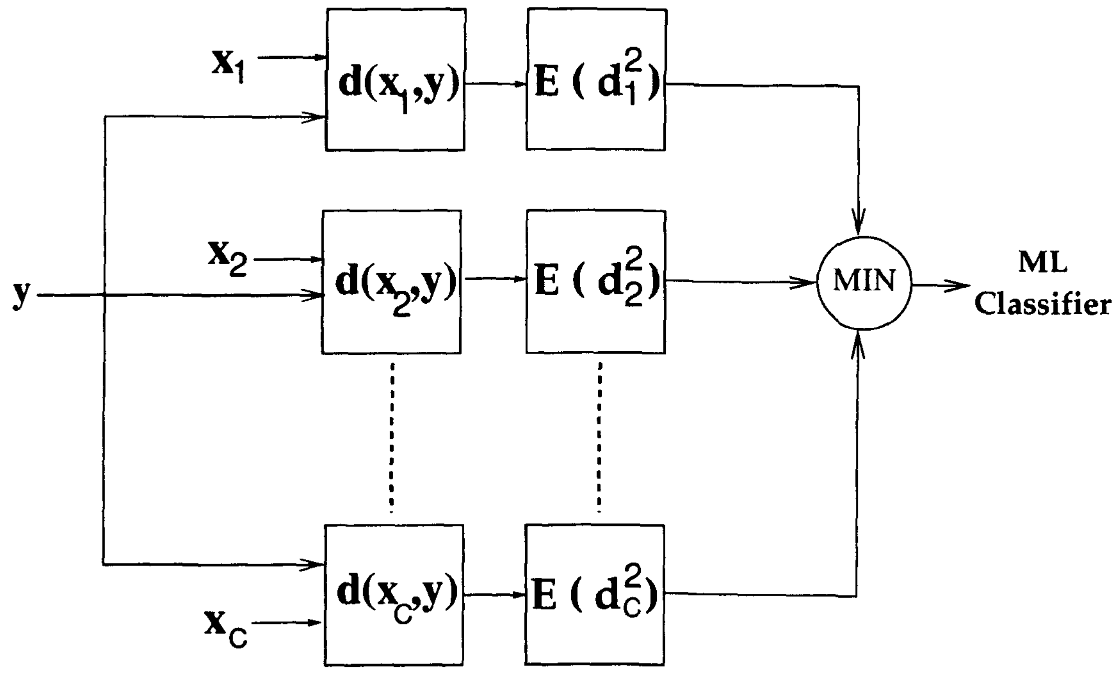


Figure 6.4: The schematic diagram for SNR estimation

$1, 2, \ldots, c$, the minimum distance (MD) modulation classifier chooses the hypothesis of which the sum squared distance function, $\mathcal{D}(\cdot,\cdot)$, is the minimum:

$$\mathcal{D}(\mathbf{Y}_k, \mathbf{I}_l) = \sum_{k=1}^{N} d(y_k, x_{li}^*) \tag{6.23}$$

with x_{li}^* and $d(\cdot,\cdot)$ defined in (6.20) and (6.21) respectively. The procedures for the MD modulation classifier is identical to the procedures for estimating the SNR described in Section 6.2.2. For this reason, the MD modulation classifier can be also be termed as minimum variance classifier.

The MD modulation classifier (shown in Figure 6.5) has reduced complexity with respect to ML modulation classifier. Besides, it requires no prior knowledge of the SNR. However, due to the fact that MD modulation classifier only takes into consideration the winning constellation point, and ignores the rest of the members in the constellation, one would expect trade-off in classification performance. The trade-off would be more obvious in lower SNR as with the case of estimation

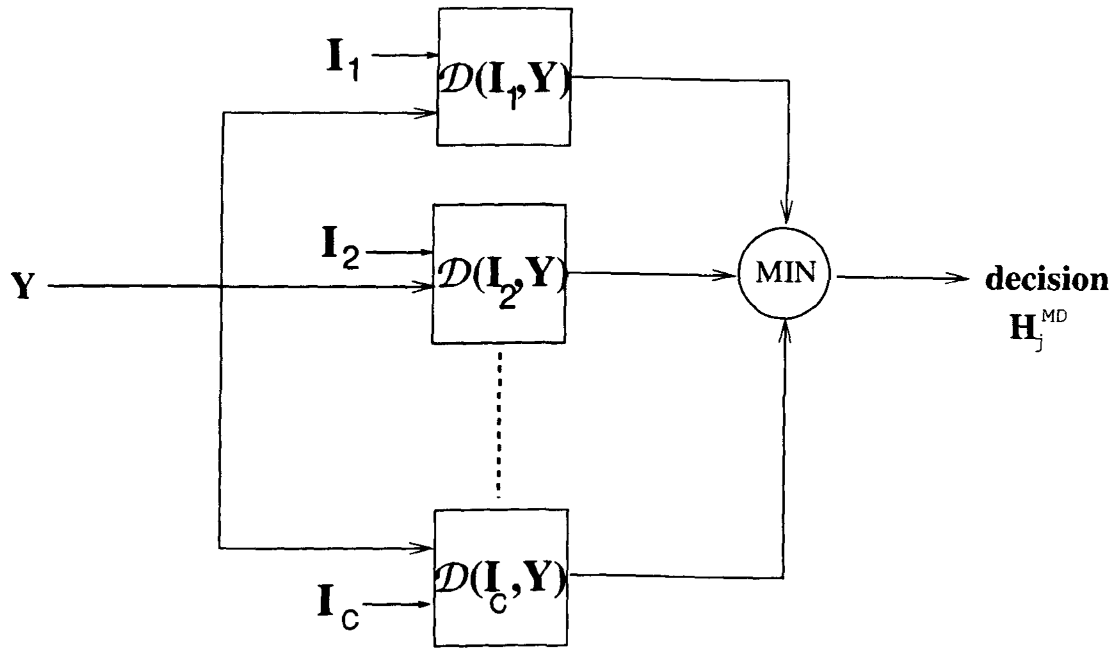


Figure 6.5: The minimum distance modulation classifier

of unknown SNR discussed in previous section. The is an interesting area of investigation for comparison between the optimal classifier and the MD classifier.

6.3 Non-coherent Maximum Likelihood Classifier

In real practical situation, the operating environment is likely to be non-ideal (c.f. (6.7)). In a non-coherent situation, we assumed that every parameter is known except for the carrier phase mismatch. The additive noise is modelled as white and Gaussian as in the coherent case. The received signal can then be modeled as the following:

$$y(n) = Ax(n)\exp(j\phi_c) + g_k(n) \tag{6.24}$$

There have been attempts in tackling the problem in non-coherent environment as reported in [34, 93, 94, 95]. Sills extended the log likelihood function to incorporate the phase information in [34] while similar approaches using likelihood ratio were reported in [94, 95] which have an extensive and in-depth study of coherent, symbol non-coherent and phase non-coherent environments, however, the modulation type were limited to M-ary PSK modulation and a hierachical aprroach is taken. In [93], A method for non-coherent BPSK and QPSK classification using an interesting antenna array method was also proposed by Hong and Ho.

The main problem of carrier non-coherency is that the received signal are presented with a translation. Geometrically it can be observed as a rotation around the origin as shown as in Figure 6.6. This rotation as can be seen from

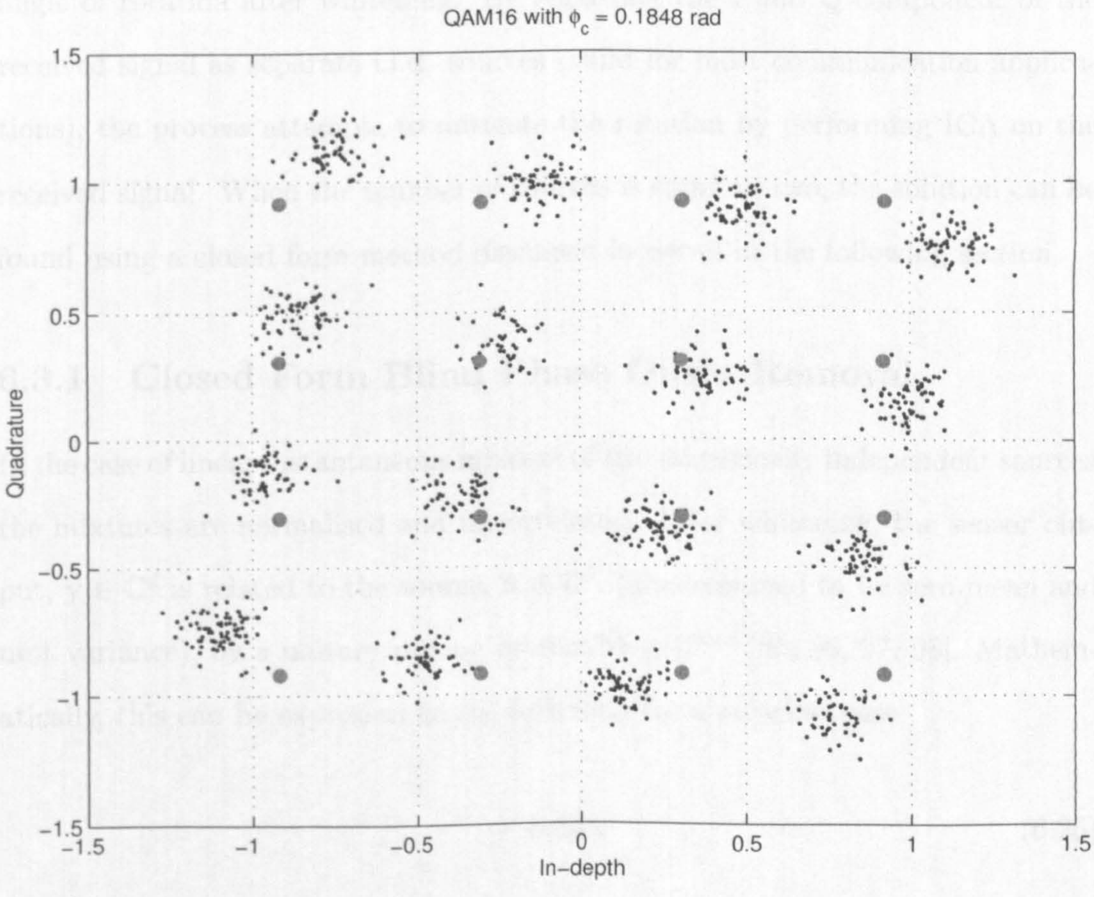


Figure 6.6: An example of received QAM16 with carrier phase offset of ϕ_c

the figure in turn causes the mismatch between the received samples and the constellation points. As a result, the previously derived likelihood function no longer holds for the non-coherent case.

As the ML method is generalised for all quadrature phase-amplitude modulation, it is therefore desirable to leave the solution as it is, and mitigating the phase mismatch problem as a separate block prior to the current classifier block. A tool that can perform blind estimation of the phase error without prior knowledge of the constellation model is required. Recalling the introduction of ICA in Chapter 5, this phenomena is similar to the two uniformly distributed unit-variance sources scenario, where performing ICA is the same as estimating the angle of rotation after whitening. By regarding the I and Q component of the received signal as separate i.i.d. sources (valid for most communication applications), the process attempts to mitigate the rotation by performing ICA on the received signal. When the number of sources is equal to two, the solution can be found using a closed form method discussed in detail in the following section.

6.3.1 Closed Form Blind Phase Offset Removal

In the case of linear instantaneous mixture of two statistically independent sources, the mixtures are normalised and uncorrelated. After whitening, the sensor output, $\mathbf{y} \in \mathbb{C}^d$ is related to the source, $\mathbf{x} \in \mathbb{C}^d$, (also assumed to be zero mean and unit variance), by a unitary mixing matrix,$\mathbf{M} \in \mathbb{C}^{d \times d}$ [86, 96, 97, 98]. Mathematically, this can be expressed as the following for a noiseless case:

$$\mathbf{y} = \mathbf{M}\mathbf{x} \tag{6.25}$$

For any $d > 2$ case, the problem can be solved by solving the elementary $d = 2$ case iteratively over the signal pairs ([98]). In this application, we treat the I and

Q components as i.i.d components ($d = 2$), which the mixing matrix (rotation matrix for whitened I and Q components) has the general shape:

$$\mathbf{M} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{6.26}$$

Given the above equation, closed form expressions for the non-iterative estimation of θ can be formulated, through the utilisation of the HOS of the signal. Furthermore, it is usually possible to find further closed form expressions to predict large sample asymptotic behaviour of these closed form estimation method.

The angle, θ, is estimated in closed form as the phase of a set of two 'centroids', which are just complex linear combinations of the 4th-order statistics of the whitened random vector. The extended maximum likelihood estimator is based on the 4^{th}-centroid [99]:

$$\begin{aligned} \xi_4 &= (\kappa_{40}^y - 6\kappa_{22}^y + \kappa_{04}^y) + j4(\kappa_{31}^y - \kappa_{13}^y) && (6.27) \\ &= \gamma \exp(j4\theta) && (6.28) \end{aligned}$$

where $\gamma = \kappa_{40}^x + \kappa_{04}^x$ is the source kurtosis sum, which can be estimated from the source using:

$$\gamma = \kappa_{40}^y + 2\kappa_{22}^y + \kappa_{04}^y \tag{6.29}$$

The angle, θ can the be estimated from the following for cases where $\gamma \neq 0$:

$$\hat{\theta}_{\text{Extended } ML} = \frac{1}{4}\angle(\xi_4\gamma) \tag{6.30}$$

For our purpose, most of the modulation types have four-fold symmetry, and therefore the kurtoses for I and Q components are therefore equal, i.e. the condition $\gamma \neq 0$ is satisfied. Consequently, the utilisation of Extended ML method is justified. The Extended ML method also has the lowest asymptotic variance among estimators from the same class.

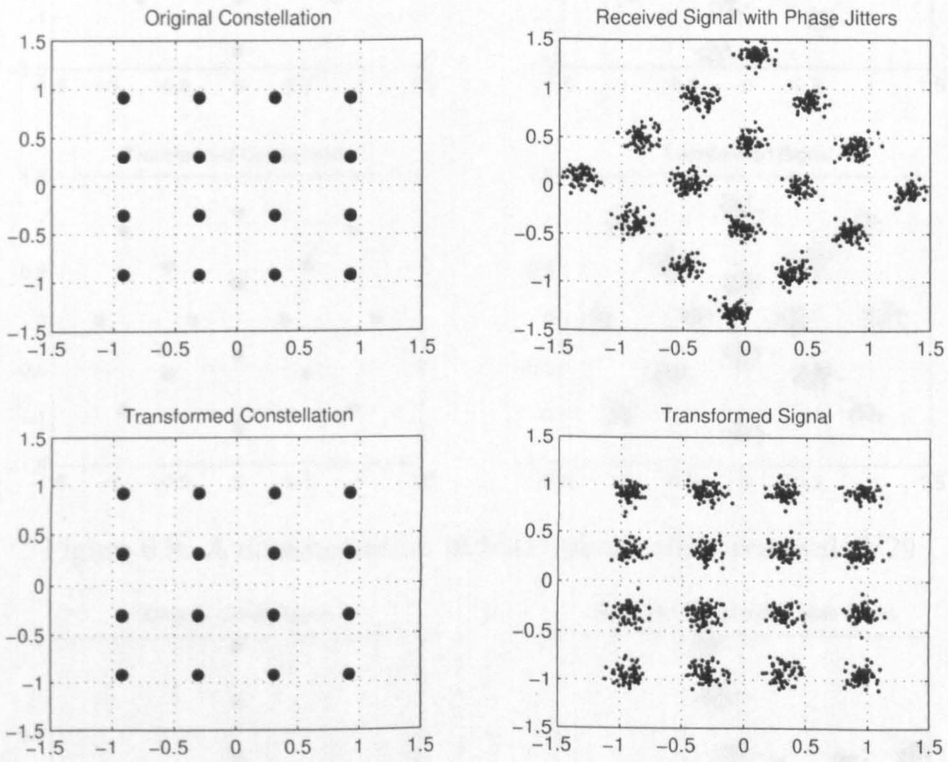


Figure 6.7: A demonstration of blind phase offset removal: QAM16

Figures 6.7, 6.8, 6.9 and 6.10 shows the result of applying the closed-form Extended ML estimator on QAM16, V.29, V.29c (Star), and QPSK modulations respectively. Note that the constellations after transformation are not always the same as the original constellations. Therefore it is proposed to apply the transform on both the original constellation and the received signal to maintain the relationship between the received sensor output and the original constellation of the modulation.

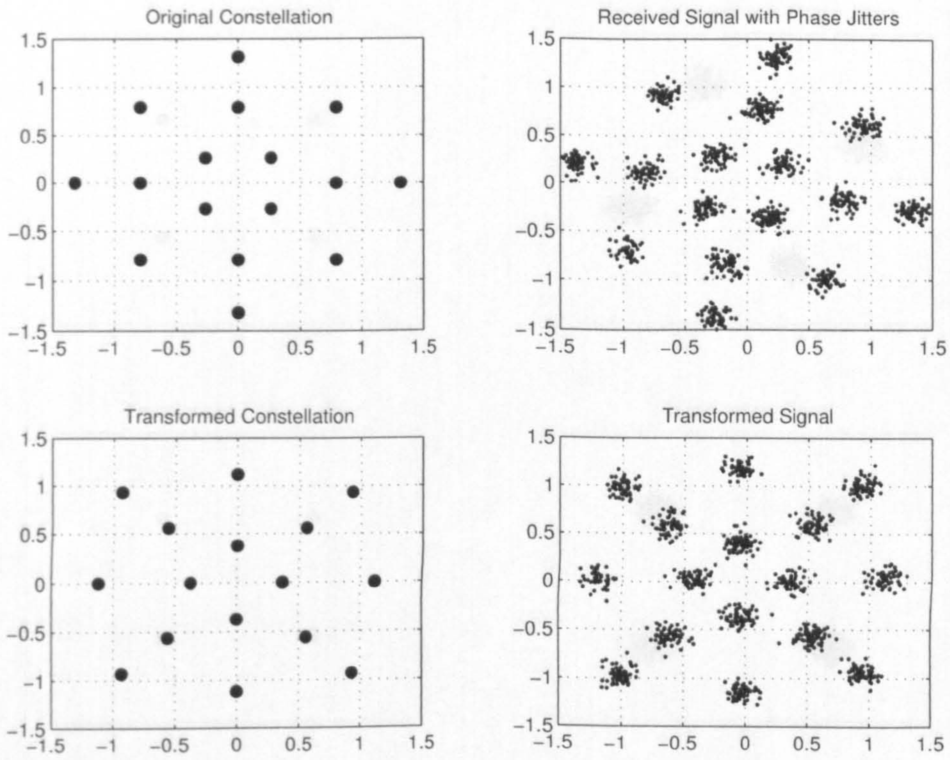


Figure 6.8: A demonstration of blind phase offset removal: V29

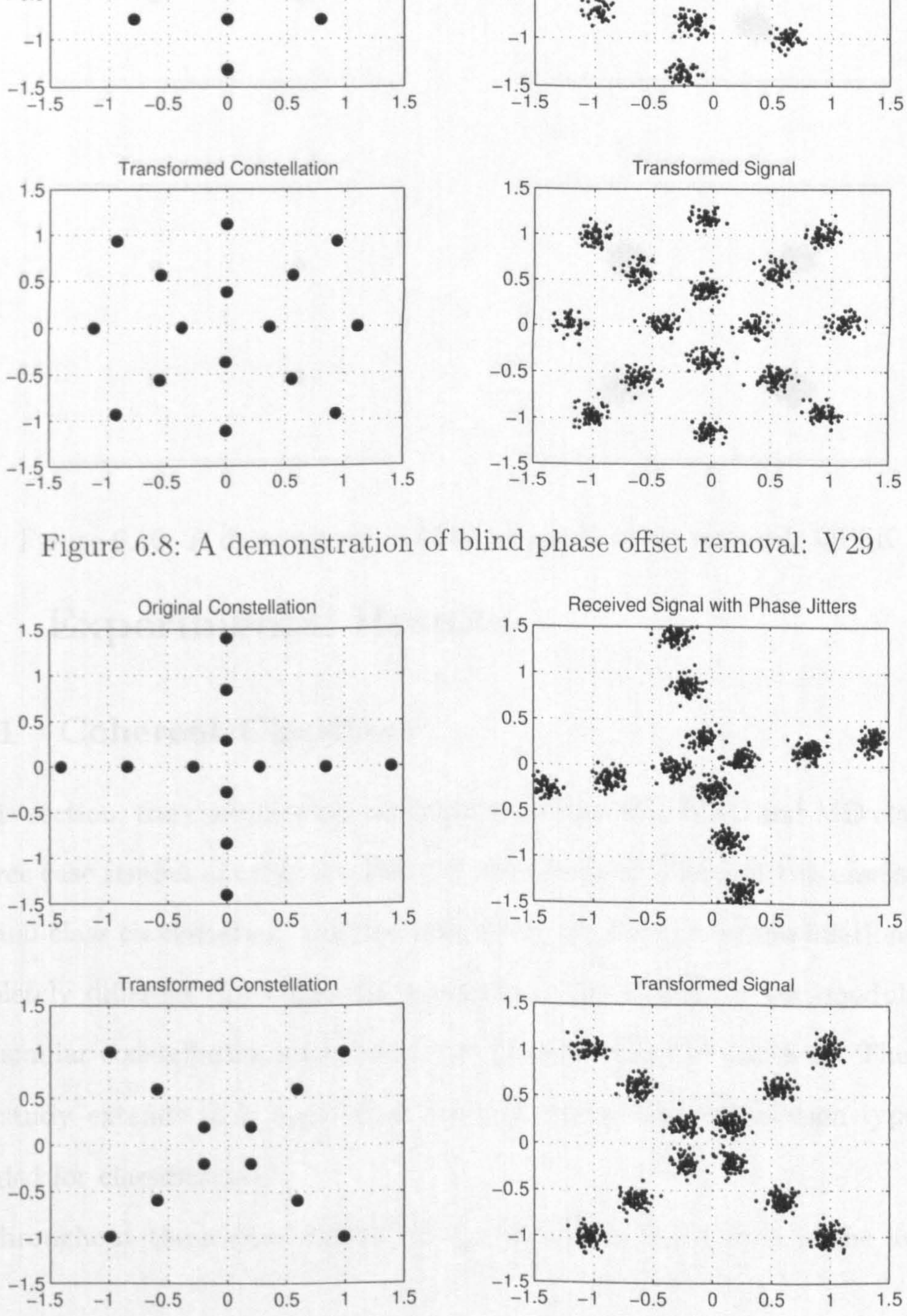


Figure 6.9: A demonstration of blind phase offset removal: V29c

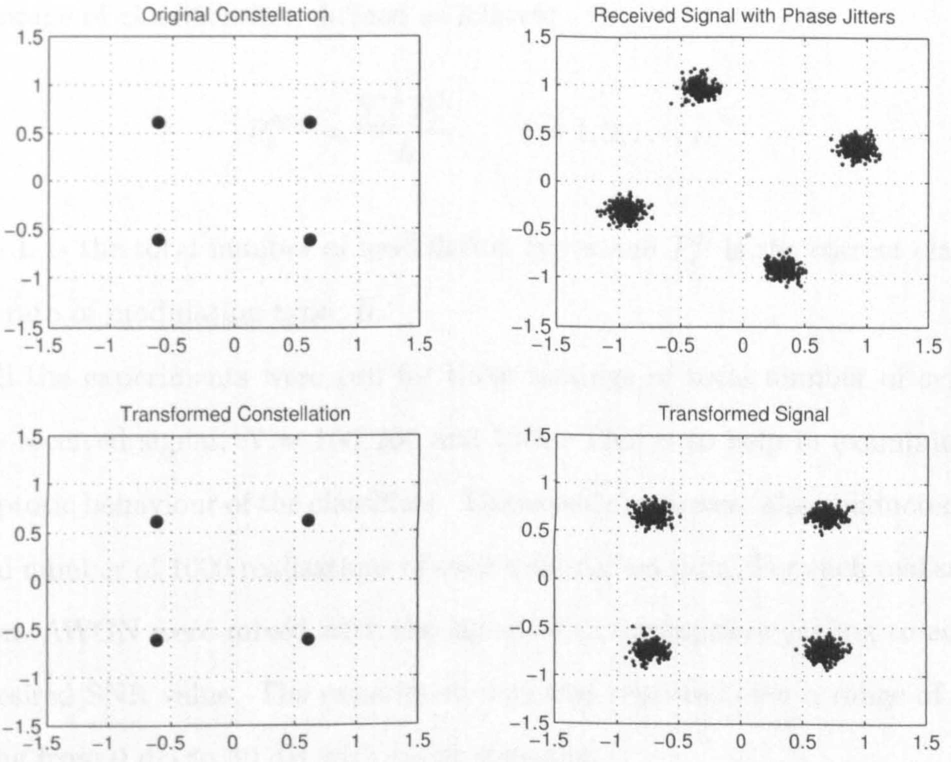


Figure 6.10: A demonstration of blind phase offset removal: QPSK

6.4 Experimental Results

6.4.1 Coherent Classifiers

In this section, the classification performance of the ML, EML and MD classifier in three case studies of coherent detection is examined. The first two case studies are dual-class classification, the first case study for the case of modulations with completely different constellations; meanwhile, the second for two modulations with similar constellation setup (with one being a subset of another). The third case study extends to a multi-class scenario where four modulation types are included for classification.

Throughout these case studies, the performance index used is the average percentage of classification, defined as follows:

$$P_c^{\text{Avg}} = \frac{\sum_l^L P_c^{I_l}}{L} \qquad l = 1, 2, \ldots, L \tag{6.31}$$

where L is the total number of modulation types and $P_c^{I_l}$ is the correct classification rate of modulation type, I_l.

All the experiments were run for three settings of total number of symbols of the received signal, $N = 100, 200$ and 1000. This is to help in examining the asymptotic behaviour of the classifiers. The experiments were also conducted with a total number of 1000 realisations of each modulation type. For each realisation, random AWGN were mixed with the signal with appropriate scaling to achieve the desired SNR value. The experiment was also repeated over a range of SNR, ranging from 0 dB to 30 dB with single stepping.

Case Study I: V.29 vs QAM16

In this case study, a dual class scenario was investigated. Two modulation types of different constellation shapes were chosen, i.e. V.29 against QAM16. In terms of shape, V.29 is said to have a diamond (or kite) shape and QAM16 is a square shape (see Figure 3.1). The distinct shape difference make it a good case for building a handle for better understanding of the behaviour of the classifiers.

The results presented here were obtained with 1000 realisations for each modulation type at each designated SNR. The experiments were run repeated from 0 dB to 30 dB SNR in step of 1 dB. At each run, random additive white Gaussian noise was added according to the desired SNR. The same received signals were fed into 3 different classifiers, i.e. the generic ML MC with actual knowledge of SNR, the EML MC with an estimated value of SNR and the MD MC. All received signal and constellation diagrams were normalised to unit energy.

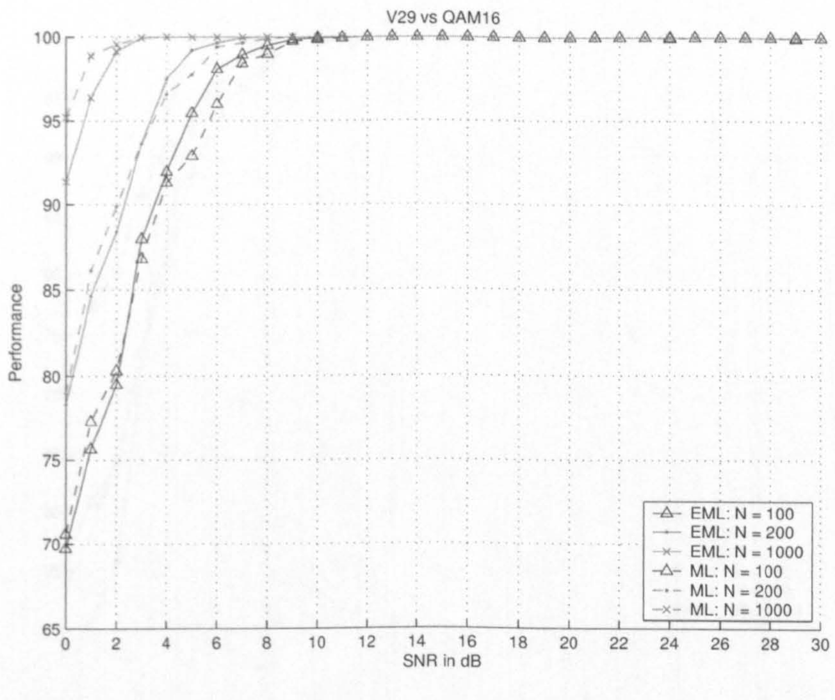


Figure 6.11: Case I: Performance of ML MC and EML MC

Figure 6.11 shows the performance of the ML MC, both generic and estimated. It was shown that the results obtained agrees with the results shown by Wang and Mendel in [32]. The figure also shows that both MCs shows good asymptotic behaviour here even at very low SNR values, where the performance increases as the sample size increases.

At lower number of samples, $N = 100$ and 200, the EML MC showed a very close match with the generic ML MC. Interestingly, the EML classifier exhibited slightly better performance at lower SNR than the generic ML MC. At first glance, it was not convincing as ML MC acts as a performance upper bound for other MCs. However, during our SNR estimation, by choosing the smallest variance estimation, we were effectively choosing the preferred modulation type (in fact the procedure is the same as doing an MD classification). This pseudo-decision implicitly suppressed other less preferred modulation types. Consequently, more information was exploited than might have appeared on the surface.

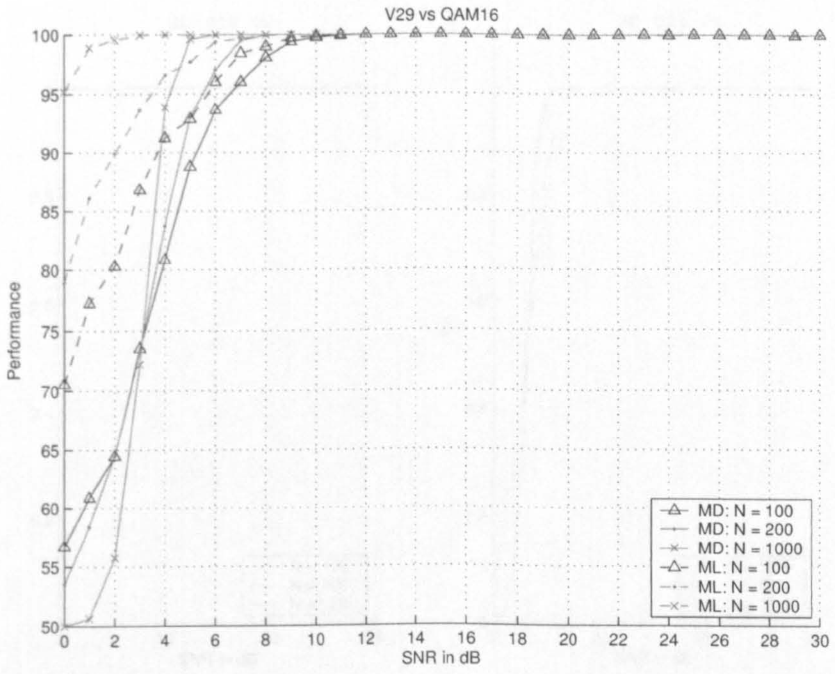


Figure 6.12: Case I: Performance of MD MC and ML MC

In Figure 6.12, the performance of the MD classifier was compared with the ML classifier. From the figures, the MD classifier shows a close match with the optimal ML classifier at high SNR value. This results supported the claim that the MD classifier are equivalent to ML classifier in AWGN environment given sufficient SNR. However, at lower SNR values, the performance is not as satisfactory. This was owing to the simplification of the MD classifier that decision was purely made using the nearest constellation point, and the relationships between the received sample and other points in the constellation were ignored.

A point worth noting was that the MD classifier showed a lower bound of 50% classification success rate at low SNR. By taking a detailed look (see Figure 6.13), it was realised that the classifier was biased to a particular modulation type at lower SNR (V.29 in this case). Therefore, the figure (50 %) did not reflect the actual scenario of what was going on.

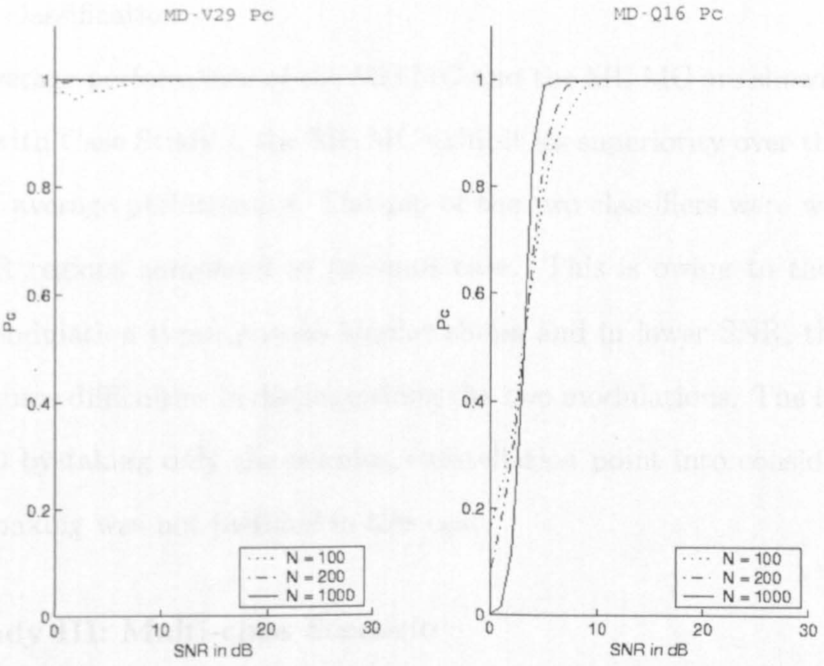


Figure 6.13: A scenario where a classifier is biased to a particular modulation type at low SNR

Case Study II: QPSK vs QAM16

The setup for the second case study was the same as the previous case study. The only difference was that two square constellations were chosen for comparison, i.e. QPSK and QAM16. Same experiments were deployed as for the previous experiments. The classification performance was again expressed with the average of the percentage of correction classification. These figures were obtained, again, from the average of 1000 examples for each modulation types.

Figure 6.14 presents the performance of the EML MC and the ML MC. The dotted line represents the performance of ML MC with appropriate legends for 100, 200 and 1000 number of samples. The solid line again shows the performance of EML MC. Once again, the two classifiers showed a close match in performance with the EML MC outperforming the ML MC by a slight margin at low SNR region. For mid and high SNRs (> 10 dB), both classifiers were achieving 100 %

of correct classification.

The average performance of the MD MC and the ML MC are shown in Figure 6.15. As with Case Study I, the ML MC exhibit its superiority over the MD MC in term of average performance. The gap of the two classifiers were wider at the lower SNR regions compared to previous case. This is owing to the fact that the two modulation types possess similar shape and in lower SNR, the effect of AWGN causes difficulties in distinguishing the two modulations. The implication of the MD by taking only the winning constellation point into consideration for decision making was not justified in this case.

Case Study III: Multi-class Scenario

The final case study of this section investigated a multi-class problem. Four modulation type were chosen as listed below:

$$I = \{\text{V.29, QAM16, V.32, V.29c (Star)}\}$$

The experiments were again conducted with different number of samples over a range of SNR as before. However, in this case study the experiments were repeated five times with 200 test examples each to get a smoother performance. The average performance was again chosen as performance index and the results are shown in Figure 6.16 and Figure 6.17.

The average correct classification performance of EML MC is shown in Figure 6.16 as solid lines and the performance of ML MC is shown as dotted lines. As with the dual class scenario, the EML MC consistently outperformed the ML MC by at lower SNRs. However the differences are not so prominent. For $N = 1000$, there was only $1dB$ difference approximately. At lower sample sizes, the differences were negligible.

Figure 6.17 shows the performance of the MD MC (solid lines) with the ML MC as reference (dotted lines). The performance difference between the two classifiers was wider than in the QPSK-QAM16 scenario. For large sample size, the performance of the MD MC is similar to that of the the ML MC at mid SNR range (SNR $\simeq$ 12 dB), but for smaller sample sizes, the SNRs needed were much higher (SNR $\simeq$ 16 dB for $N = 200$ and SNR $>$ 20 dB were needed for $N = 100$).

6.4.2 Non-Coherent Classifiers

In this final section of the chapter, the performance of the classifiers in the non-coherent environment with the aid of the proposed closed form blind phase removal tool is examined. The experimental setup is the same as the previous experiments, with the exception that, for each received sample a constant phase offset is added to the received signal simulating the unknown phase mismatch, θ_c.

$$-\frac{\pi}{2} \leq \theta_c \leq \frac{\pi}{2} \tag{6.32}$$

We assume high sampling rate so that the carrier phase offset is constant throughout the length of the received signals.

Case Study IV: Non-coherent performance of the proposed classifiers

As in previous case studies, Monte Carlo simulations were conducted for SNRs ranging from 0 dB to 30 dB. At each SNR, 200 examples of received signal were mixed with AWGN and random phase offsets. This was repeated for 5 trials with 3 different number of samples at each trials, $N = 100, 200$ and 1000. Four modulation types were again chosen for multi-class classification. The four digital modulation types were the same as those used for Section 6.4.1. The results obtained for EML MC, ML MC and MD MC were presented in Figure

6.18, Figure 6.19 and Figure 6.20 respectively.

The performances of the EML classifier with the aid of the closed form phase removal tool are shown in solid lines while the performances of its coherent counterpart without the tool are drawn using dotted lines. At $N = 100$, the performance of the non-coherent case is about 5% less than the coherent case. However, it is still in the high ninety percent for correct classification for high SNRs. For higher values of N, the performance of the noncoherent EML MC equated the performance of the coherent EML MC at high SNR. Nevertheless, the performances were typically $5 \sim 10\%$ lower in low SNR regions.

In Figure 6.19, the same setup were adopted for presenting the results of the ML classifier in non-coherent environment and coherent environment. A close match of performance was observed for the high SNR region regardless of the the sample size. At sample size of a hundred, the performances of both scenario converged to almost the same percentage of correct classification. The same fact can be observed at higher samples size. At lower SNR, it was observed that more trials was needed for a better smoothness of the performance.

Lastly, the performances of the MD MC were examined for the non-coherent environment in Figure 6.20. As with the previous results, the performances showed close matching in all SNRs, with the non-coherent case lagged with ~ 1 dB in performance for lower SNRs. As we have seen in previous cases, the SNR needed for MD MC is more than the ML methods of modulation classification.

6.5 Conclusion

The ML approach has advantages over the conventional hypothesis testing and ANN based methods. In hypothesis testing, the order of tests for multi-class modulation classification can be influential to the final classification results. Repetitive testing is needed for determination of key features thresholding. The hypothesis testing method has its limitation in extending the algorithm. ANN based method on the other hand, faces the difficulty in getting training examples for statistical reasons. Besides, the hassle of re-training the ANN for different SNRs appears as another drawback for ANN based classifier.

To these ends, the ML approach has major advantage over these methods. The method exploits the probabilistic relation between the received samples and is derived in accordance with classical detection theory [100]. There is no extraction of key features, hence no extra effort required for determination of thresholds for different modulation types. The prior knowledge needed is nothing more than the known constellations of the modulation types. Assuming that the operating environment is AWGN with others sufficient treatment for the channel, the ML is the optimal classifier for method of the same class.

We developed a simple SNR estimation method in complement to the ML MC for the case when SNR is not known *a priori*. Experimental results show that this method works very well, and in some cases, it had slightly better performance than generic ML. This is believed to be caused by the fact that the EML MC is implicitly a cascaded MD and ML classifier. The MD MC was also derived from the ML MC without the exploitation of the knowledge of SNR. The performance of MD MC was shown through experiment to be not as good as ML MC in low SNR regions, but it offered an option with less computational complexity.

A novel application of ICA was deployed in dealing with the non-coherent environment where a phase offset was introduced to the coherent signal model. The method was based on the fact the capability of ICA in compensating the angle of rotation of the constellation caused by the phase mismatch. A closed form solution method was adopted and the results were close to those obtained for coherent environment.

In short, ML based methods offer a realistic, simple and extendible solution preferred to the conventional hypothesis testing as well as the newer ANN based pattern recognition approach. The case studies proved that the ML based methods are robust and give good performances.

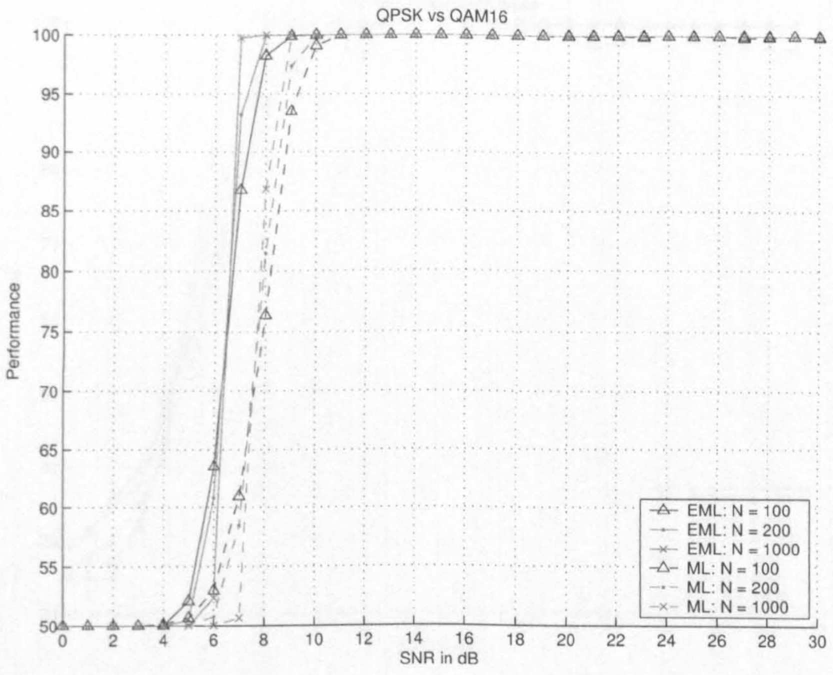


Figure 6.14: Case II: Performance of the EML MC and the ML MC

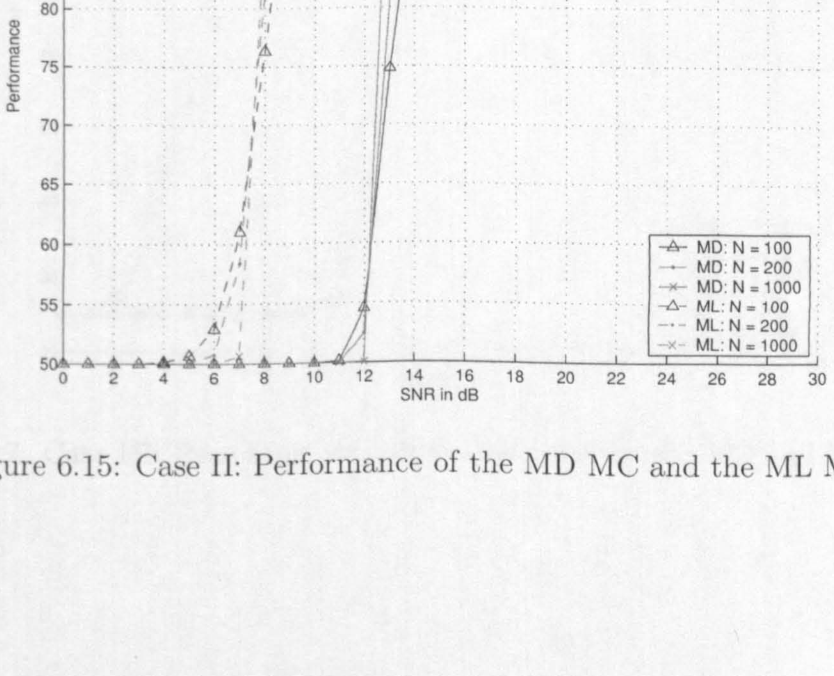


Figure 6.15: Case II: Performance of the MD MC and the ML MC

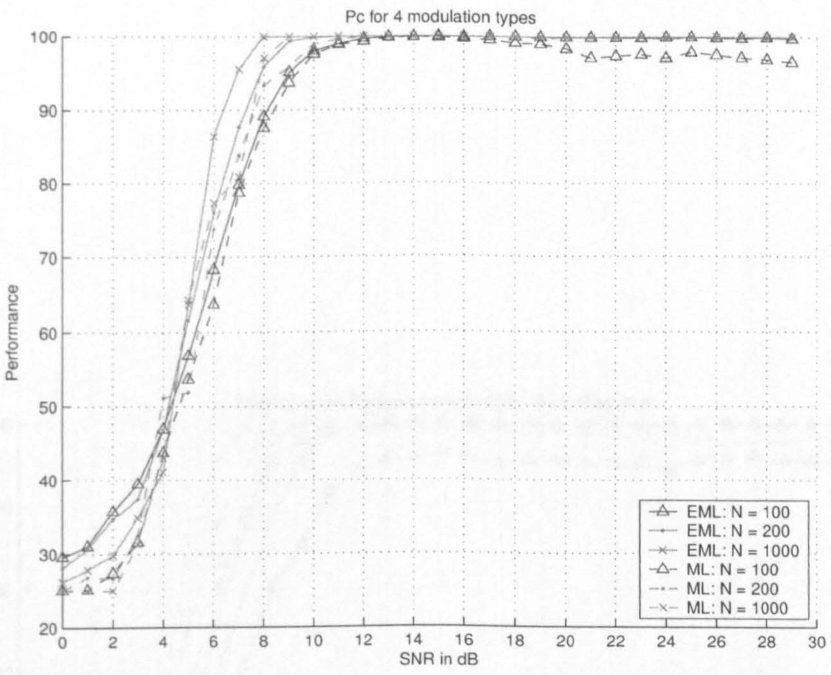


Figure 6.16: Case III: Percentage of correct classification for EML and ML classifiers

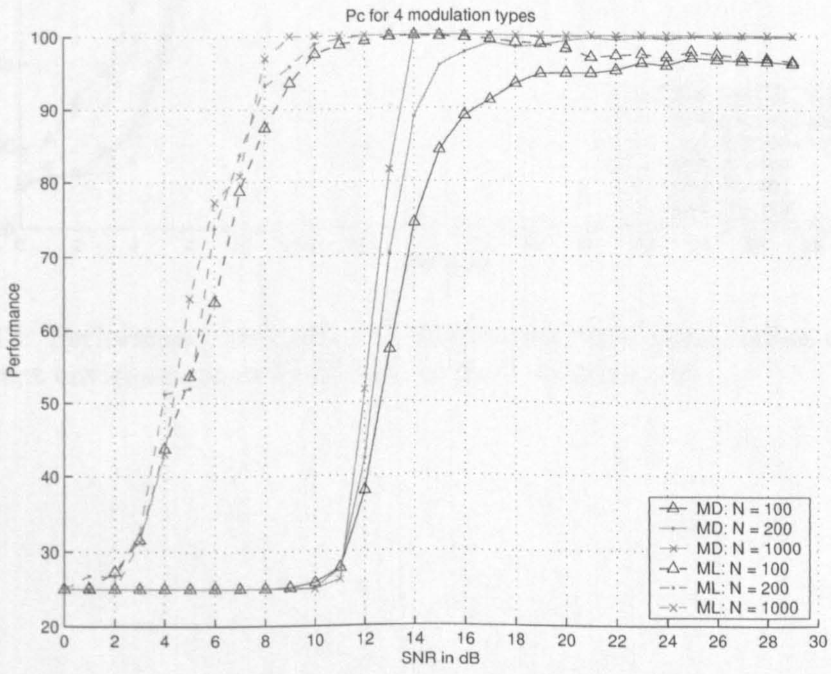


Figure 6.17: Case III: Percentage of correct classification for MD and ML classifiers

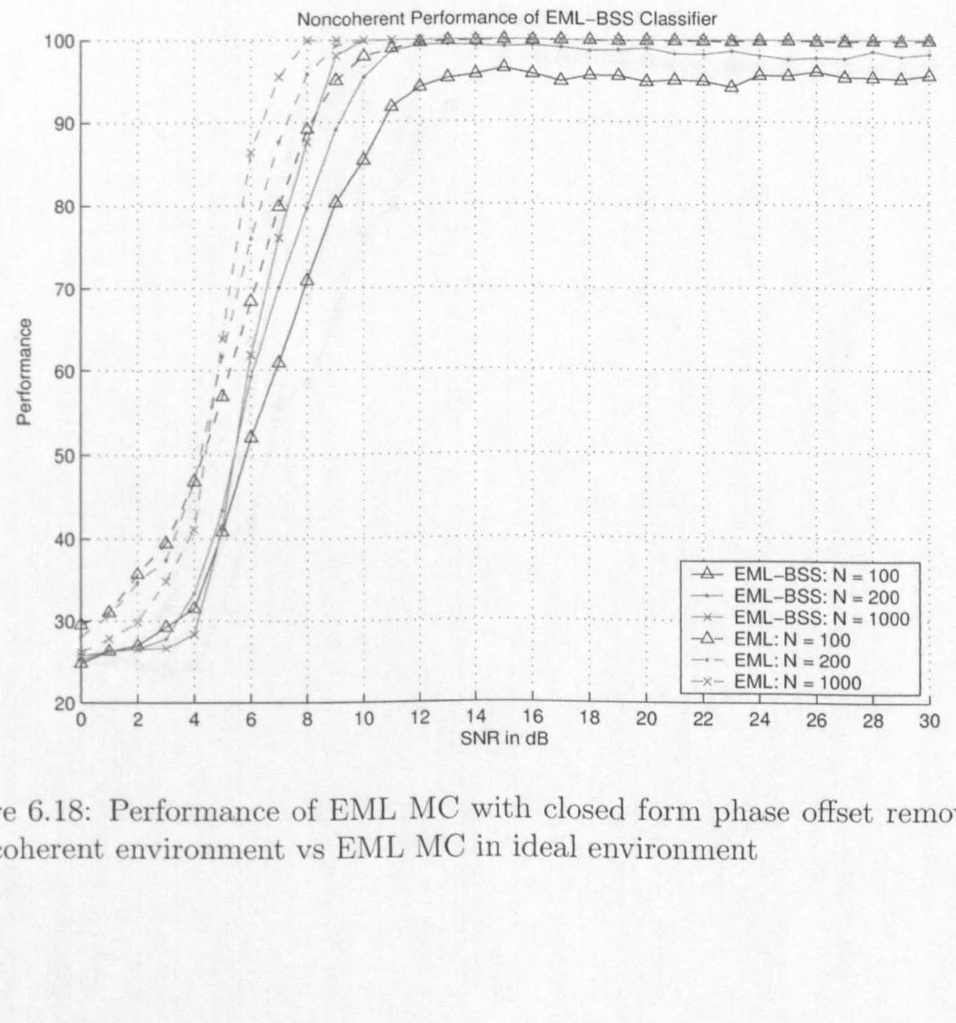


Figure 6.18: Performance of EML MC with closed form phase offset removal in non-coherent environment vs EML MC in ideal environment

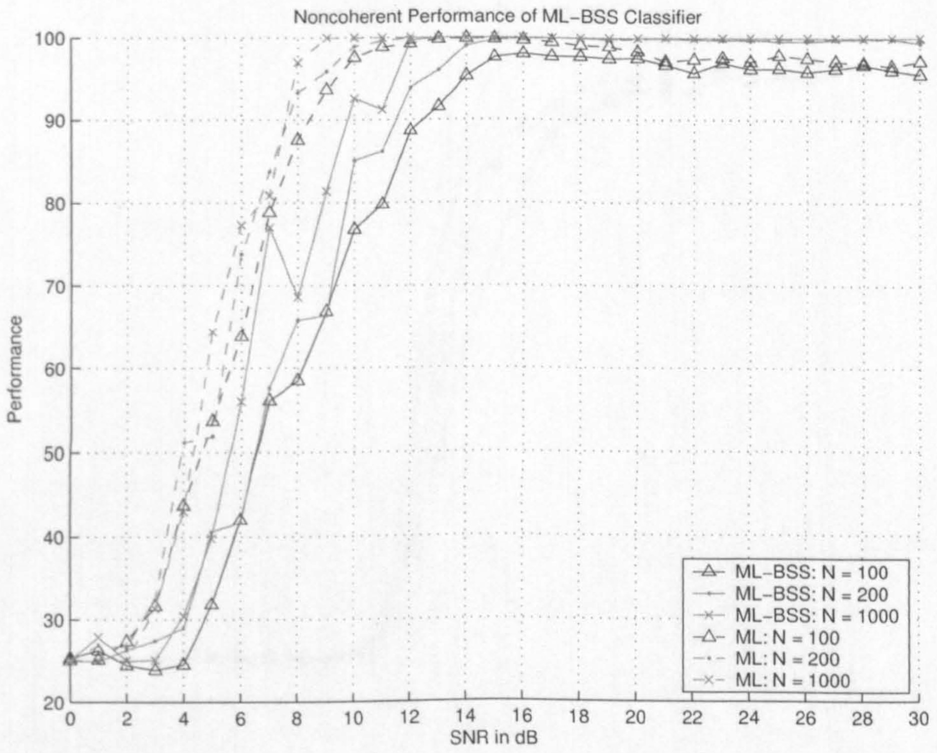


Figure 6.19: Performance of ML MC with closed form phase offset removal in non-coherent environment vs ML MC in ideal environment

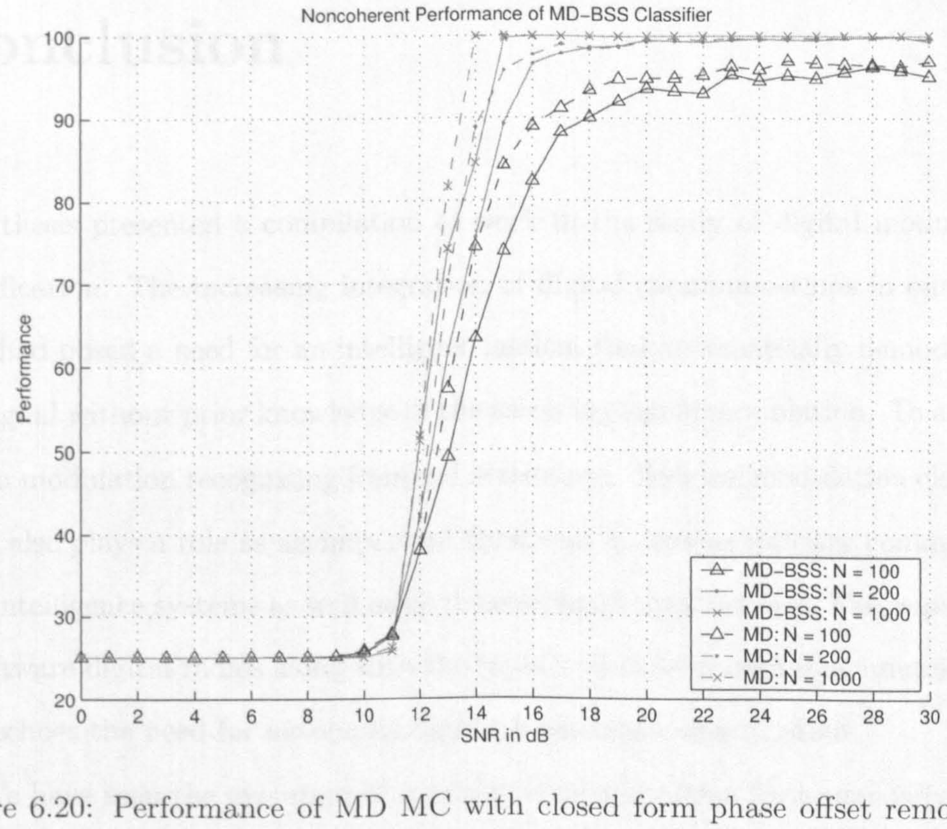


Figure 6.20: Performance of MD MC with closed form phase offset removal in non-coherent environment vs MD MC in ideal environment

Chapter 7

Conclusion

This thesis presented a compilation of work in the study of digital modulation classification. The increasing integration of digital communications in our daily lives had posed a need for an intelligent modem that automatically demodulates the signal without prior knowledge of the incoming signal modulation. To achieve this, a modulation recognising frontend is required. Besides, modulation classification also plays a role as an important front end in various military communication intelligence systems as well as civil surveillance applications. The popularity of software digital radios along with the rapid evolution of digital communication also echoes the need for automatic digital modulation classification.

We have seen the evolution of modulation classification from manually operated system to nowadays abundant choices of fully automatic systems (Chapter 2). The study of automatic classification of signal modulation has gone a long way from it first began. With a firmer grip of more advanced signal processing and pattern recognition techniques, researchers have been able to improve the classification performance through these years. The proposal of ANN for modulation classification presents a elegant and robust substitute to the conventional decision theoretics methods. A wide choice of different variants of ANNs also give the user customised solutions depending on the circumstances. While ANN methods lean toward a more practical investigation of the problem, ML based solutions offer better theoretical value.

We summarised the major contributions of this work on a chapter by chapter basis in the following section. A broad view of considerations and comments for future direction of this research are discussed in concluding this work.

7.1 Summary by Chapters

The backgrounds and literature survey of the field were given at the start of this work. The literatures surveyed includes the originating work of automatic modulation recognition as well as many recent contributions by various researchers. Some necessary pre-requisites were also included at the end of Chapter 2.

The mathematical treatments of the problem were laid out in the subsequent chapter (Chapter 3). A novel higher order statistical feature set was introduced in this chapter. The feature set (consisting of only three HOS based features) was designed based on the fact that each modulation had a unique constellation signature under the coherent environment. The higher order cumulants capture the statistical descriptions of the in-phase and quadrature components of the complex envelope of the received signal. The variants of the proposed feature set for various SNR were also found through numerical computation.

In Chapter 4 of the thesis, three variants of supervised neural networks (MLP, RBF network and PNN) were investigated. These neural networks were examined in terms of classification performance via computer experiments. The RPROP MLP excelled among them, giving a high classification performance of 98% in high SNR while giving 50 % of classification accuracy at 0 dB SNR. This was remarkable considering the broad range of digital modulations tested in this chapter, 12 modulation types in total. The classifying system also showed a good asymptotic behaviour in terms of classification performance. The PNN, being another variant of the neural network, proved to be a strong contender to the RPROP MLP in term of classification performance. Besides, it offered a shorter training time along with other advantage over the BP training based MLP classifiers. RBF network based classifier was also investigated in this chapter. Experimental results were also presented at the end of the chapter.

There have been a broad collection of feature set proposed by various researchers for the problem of modulation classification. Chapter 5 discussed the issues of feature selection and transformation. The primary aim of feature selection was to reduce the input dimension to the classifier by choosing the most effective feature subset. The author proposed the use of GA for feature selection and two methods of such algorithm to meet different design requirements. The first approach, using a binary string genome, dynamically searched the whole feature space. However, the user loses control on the final reduced input dimension, as this is controlled by the GA. The latter approach, list string genome, rectified this drawback by including an extra parameter to fix the exact number of features needed. By using the latter approach, a performance of $\sim$ 96 % of correct classification was recorded for 0 dB SNR using only 3 features out of the total of 17 features. The chapter continued to explored another type of dimension reduction method based on linear transformation matrix. Two method were investigated again, namely PCA and ICA. From the experimental results, it was shown that ICA, although an extension of PCA, did not outperform the PCA. Both method recorded similar performanes, giving more than 90% across all SNRs using 9 transformed features, and over 70 % across all SNRs using only 2 features. As the dataset used in this chapter was different from the previous chapter, much to the author's regret, direct comparison was not possible.

In the final part of this work (Chapter 6), an alternative approach to the neural network approach and the decision theoretic approach, the maximum likelihood approach was introduced. The introduction began with the discussion of The law of total probability and Bayesian rule for classification. Subsequently, the ML classifier were derived for the coherent case of operating environment. A more practical implementation of the ML classifier using a novel yet trivial SNR estimation technique was introduced. From the ML classifier, a new minimum distance classifier was derived. The performance of these three classifier was reviewed for the dual class scenario and the multi class scenario involving 4 different modulation types. As the ML classifier is only optimal in the coherent environment, we proposed the novel utilisation of a blind phase correction tool: the closed form extended maximum likelihood BSS estimator. The performance of the classifiers with the aid of the closed form estimator under non-coherent environments were also investigated. Selected results were given at the end of this chapter.

In conclusion, the supervised neural networks promised excellent performance without any fuss of threshold determination. The novel HOS based features was proven to be effective and robust, particularly in high SNRs under coherent scenarios. When there exists a large number of features or when the characteristics of the features were not adequately understood, feature selection and transformation could be used in refining the performance of the neural classifiers. When training examples are not accessible to the user, ML approach is a strong alternatives to the neural classifier. Using the blind closed form estimator, the ML approach and its variants can be applied directly without any alteration in likelihood function. The classification performance obtained via this method suffered little degradation.

7.2 Future Directions

The author suggests the following for consideration of the extension of the existing works:

1. The design and investigation of a single neural classifier for all SNRs. One particular advantage of the ML approach is that a single global solution for all SNR. This is not the case for neural classifier at the moment, it would be a desirable feature if there exists an universal neural classifier.

2. The investigation of a adaptive scale factor for PNN classifier. As seen from the results obtained in Chapter 2, it hints that the optimal scale factor seems to change with the SNRs. A possible method is determination of scale factors from the statistics of the training dataset.

3. Extension of the investigation of ML approach to more modulation type. The proposed method seemed to be robust but currently the investigation is limited to few modulation types only.

4. A unified dataset to facilitate the direct performance comparison between the ANN classifiers and ML classifiers.

5. The implementation of current classifiers (Neural and ML) on to real time DSP based solution. This will enable real field application of the algorithms. Integration of the algorithms to a practical active software radio project would also be of great interest e.g. Free Software Foundation (FSF) GnuRadio Project [101].

6. The investigation of other framework for modulation classification e.g. the phase lock loop based method [102, 103], fuzzy logic frame work [33], etc.

7. Feature selection and transformation are only required under specific circumstance where dimension reduction is required. However, these are interesting and practical problems, investigation should continues with some recently proposed approach such as the maximum representative and discriminative features (MRDF) transformation [104]. Genetic programming is also proposed to be a quicker alternative to GA in feature selection.

Appendix A

The FastICA Algorithm

In this thesis, the FastICA algrotihm was used in evaluating the application of ICA in feature transformation. The algorithm is available freely in MATLAB and R software packages. The FastICA website can be accessed through the following URL:

- `http://www.cis.hut.fi/projects/ica/fastica`

Tables A.1 and A.2 detail the algorithm for a single IC and mulitple ICs respectively. These tables are included here to facilitate the relevant discussion in this thesis and for easier reference for interested readers.

In both Table A.1 and Table A.2, the statistical expectation $E\{\cdot\}$ is approximated by sample average. A parallel version of FastICA that uses symmetric decorrelation exists. However it was omitted here, as it was not investigated in this work. The version of FastICA adopted was as described in Table A.2. The readers are referred to [105, Chapter 8] for further information and detailed mathematical derivation of the algorithm using negentropy.

1. Remove the mean from the data.
2. Whiten the data to give $\mathbf{z}$.
3. Initialise a unit norm random $\mathbf{w}$ vector.
4. Let $\mathbf{w} \leftarrow E\{\mathbf{z}g(\mathbf{w}^T\mathbf{z})\} - E\{\mathbf{z}g'(\mathbf{w}^T\mathbf{z})\}\mathbf{w}$, where g is a chosen non-linearity function e.g. *tanh*.
5. Let $\mathbf{w} \leftarrow \mathbf{w}/\|\mathbf{w}\|$.
6. Check for convergence, repeat step 4 otherwise.

Table A.1: A brief summary of the FastICA algorithm for one IC

1. Remove the mean from the data.
2. Whiten the data to give $\mathbf{z}$.
3. Choose m, the number of ICs to estimate, and initialise the counter $p \leftarrow 1$.
4. Initialise a unit norm random $\mathbf{w}_p$.
5. Let $\mathbf{w_p} \leftarrow E\{\mathbf{z}g(\mathbf{w}_p^T\mathbf{z})\} - E\{\mathbf{z}g'(\mathbf{w}_p^T\mathbf{z})\}\mathbf{w}$, where g is a chosen non-linearity function e.g. *tanh*.
6. Execute the orthogonalisation:

$$\mathbf{w}_p \leftarrow \mathbf{w}_p - \sum_{j=1}^{p-1}(\mathbf{w}_p^T\mathbf{w}_i)\mathbf{w}_j \tag{A.1}$$

7. Let $\mathbf{w}_p \leftarrow \frac{\mathbf{w_p}}{\|w_p\|}$.
8. Check for convergence, repeat step 5 otherwise.
9. increase counter p, if $p \leq m$, repeat step 4.

Table A.2: A summary of the FastICA algorithm for multiple ICs

References

[1] Perry M. Mistry, "Third generation cellular (3G): W-CDMA & TD-CDMA," in *Proc. WESCON'98*, Anaheim, CA, September 1998, pp. 227–231, Wescon.

[2] R. Prasad, *CDMA for wireless personal communications*, Artech House, 1996.

[3] Andrew J. Viterbi, *CDMA: Principles of spread spectrum communication*, The Addison-Wesley Wireless Communications series. Addison-Wesley Pub. Co., Boston, 1995.

[4] International Telecommunication Union, "ITU actvities on IMT-2000," Internet,http://www.itu.int/home/imt.html.

[5] Luca Becchetti, Petri Mähönen, and Luis Munoz, "Enchancing ip service provision over heterogeneous wireless networks: A path toward 4g," *IEEE Commun. Mag.*, vol. 39, no. 8, pp. 74–81, 2001.

[6] Manuel Dinis and Jose Fernandes, "Provision of sufficient transmission capacity of broadband mobile multimedia: A step toward 4g," *IEEE Commun. Mag.*, vol. 39, no. 8, pp. 74–81, 2001.

[7] Thierry Turletti, "Software Radio Resource Page," Internet, http://www-sop.inria.fr/rodeo/personnel/Thierry.Turletti/SoftwareRadio.html.

[8] Mahmoud Naghshineh, Ed., *IEEE Personal Communications: Special Issues on Software Radios*, vol. 6, IEEE Communications Society, 1999.

[9] Enrico Burachini, "The software radio concept," *IEEE Commun. Mag.*, vol. 38, no. 9, pp. 138–143, 2000.

[10] F. F. Liedtke, "Computer simulation of an automatic classification procedure for digitally modulated communication signals with unknown parameter," *Signal Processing*, vol. 6, pp. 311–323, 1984.

[11] L. Yingwei, N. Sundararajan, and P. Saratchandran, "Identification of time-varying nonlinear systems using minimal radial basis function neural networks," *IEEE Proc. Control Theory*, vol. 144, no. 2, pp. 202–208, 1997.

[12] F. L. Lewis, S. Jagannathan, and A. Yesildirek, *Neural Network Control of Robot Manipulators and Nonlinear Systems*, chapter Neural Network Control of Nonlinear Systems, pp. 277–304, Taylor and Francis, 1999.

[13] D. E. Goldberg, *Genetic Algorithms: in search optimisation and machine learning*, Addison-Wesley, 1989.

[14] D. E. Goldberg and M. P. Samtani, "Engineering optimization via genetic algorithm," in *Proc. 9th Conf. on Electronic Computation*, 1986, pp. 472–482.

[15] C. Ergn and K. Hacioglu, "Multiuser detection using a genetic algorithm in CDMA communications systems," *IEEE Trans. Commun.*, vol. 48, pp. 1374–1383, Aug. 2000.

[16] T. Takeuchi, Y. Nagai, and Y. Enomoto, "Fuzzy control of a mobile robot for obstacle avoidance," *Information Sciences*, vol. 45, no. 2, pp. 231–248, 1988.

[17] E. E. Azzouz and A. K. Nandi, *Automatic Modulation Recognition of Communication Signals*, Kluwer Academic Publishers, 1996.

[18] E. E. Azzouz and A. K. Nandi, "Procedures for automatic recognition of analogue and digital modulations," *IEE Proc. Comms*, vol. 143, no. 5, pp. 259–266, 1996.

[19] A. K. Nandi and E. E. Azzouz, "Algorithms for automatic modulation recognition of communication signals," *IEEE Trans. Commun.*, vol. 46, no. 4, pp. 431–436, 1998.

[20] D. Donoho and X. Huo, "Large-sample modulation classification using hellinger representation," *Signal Processing Advances in Wireless Communications*, pp. 133–137, 1997.

[21] C. Louis P. Sehier, "Automatic modulation recognition with a hierarchical Neural Network," in *Proc. MILCOM'93: Communications on the move*, Beijing, China, 1993, IEEE, vol. 1, pp. 111–115.

[22] Lu Mingquan, Xiao Xianci, and Li LeMing, "Cyclic spectral features based modulation recognition," in *Proc. Int. Conf. Commun. Tech.*, Beijing, China, 1996, vol. 2, pp. 792–795.

[23] Lu Mingquan, Xiao Xianci, and Li Lemin, "AR modeling based features extraction for multiple signals for modulation recognition," in *Proc. 4th Int. Conf. Signal Processing*, 1998, vol. 2, pp. 1385–1388.

[24] A. K. Nandi and E. E. Azzouz, "Modulation recognition using artificial neural networks," *Signal Processing*, vol. 56, no. 2, pp. 165–175, 1997.

[25] P Mackenzie, L Doyle, D O'Mahony, and K Nolan, "Modulation scheme

recognition techniques for software radio on a general purpose processor platform.," *Proc. 1st Joint IEI/IEE Symp. Telecommunication Syst.*, 1999.

[26] Communications Research Centre Canada, "Spectrum Explorer Software," Internet,http://www-ext.crc.ca/spectrum-explorer/spectrum-explorer6/en/index.htm, 2002.

[27] D. Boudreau, C. Dubuc, F. Patenaude, M. Dufour, J. Lodge, and R. Inkol, "A fast modulation recognition algorithm and its implementation in a spectrum monitoring application," in *Proc. MILCOM'00.* October 2000, vol. 2, pp. 732–736, IEEE.

[28] Ananthram Swami and Brian M. Sadler, "Hierachical digital modulation classification using cumulants," *IEEE Trans. Commun.*, vol. 48, no. 3, pp. 416–429, 2000.

[29] Bijan G. Mobasseri, "Digital modulation classification using constellation shape," *Signal Processing*, vol. 80, pp. 251–277, 2000.

[30] M. L. D. Wong and A. K. Nandi, "Automatic digital modulation recognition using spectral and statistical features with multi-layer perceptrons," in *Proc. ISSPA'01.* ISSPA, August 2001, vol. II, pp. 390–393.

[31] M. L. D. Wong and A. K. Nandi, "Automatic digital modulation recognition using artificial neural network and genetic algorithm," *Signal Processing*, vol. 84, no. 2, pp. 351–365, 2004.

[32] Wen Wei and Jerry M. Mendel, "Maximum-Likelihood Classification for Digital Amplitude-phase Modulation," *IEEE Trans. Commun.*, vol. 48, no. 2, pp. 189–193, 2000.

[33] Wen Wei and Jerry Mendel, "A fuzzy logic method for modulation classification in nonideal environments," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 3, pp. 333–344, June 1999.

[34] J A Sills, "Maximum-likelihood modulation classification for PSK/QAM," in *Proc. of MILCOM'99*. IEEE, 1999, pp. 217–220.

[35] K. Nolan, L. Doyle, P. Mackenzie, and D O'Mahony, "Modulation scheme classification for 4g software radio wireless network," in *Proc. IASTED Int. Conf Signal Processing, Pattern Recognition, and Applications SPPRA'02*, 2002.

[36] S. Haruyama, R.Morelos-Zaragoza, and Y. Sanada, "A software-defined radio platform with direct conversion: Soprano," *submitted to Kluwer Journal on Wireless Personal Communications, Special Issue on Wireless Personal Multimedia Communications.*, 2002.

[37] Simon Haykin, *Communication Systems*, John Wiley and Sons Inc., 4 edition, 2001.

[38] Robert N. McDonough and Anthony D. Whalen, *Detection of Signals in Noise*, Academic Press, 24-28 Oval Road, London NW1 7DX, second edition, 1995.

[39] Sir Maurice Kendall and Alan Stuart, *The advanced theory of statistics*, vol. 1, Charles Griffin & Company Limited, 4th edition, 1977, QA276A2K31.4(v.1).

[40] Gilles Burel, Andre Quinuis, and Stephane Azou, "Interception and furtivity of digital communication transmission," in *Invited paper, Proc. IEEE Communications 2002*, Bucharest, Romania, Dec 2002.

[41] Thomas Keller and Lajos Hanzo, "Adaptive multicarrier modulation: A convenient frame work for time-frequency processing in wireless communications," *IEEE Proc.*, vol. 88, no. 5, pp. 611–639, 2000.

[42] K. E. Nolan, L. Doyle, D. O'Mahony, and P Mackenzie, "Signal space based adaptive modulation for software radio," in *Proc. IEEE Wireless Communications and Networking Conf.* IEEE, March 2002, pp. 510–515.

[43] M Luise and R. Reggiannini, "Carrier frequency recovery in all-digital modems for burst mode transmissions," *IEEE Trans. Commun.*, vol. 43, pp. 1169–1178, 1995.

[44] Tyler Brown and Michael Mao Wang, "An interative algorithm for single-frequency estimation," *IEEE Trans. Signal Processing*, vol. 50, no. 11, pp. 2671–2682, Novemeber 2002.

[45] A. K. Nandi, Ed., *Blind Estimation Using Higher Order Statistics,* Kluwer Academic Press, London, 1999.

[46] John J Shynk and Christina K Chan, "Performance surfaces of the constant modulus algorithm based on a conditional Gaussian model," *IEEE Trans. Signal Processing*, vol. SP-41, no. 5, pp. 1965–1969, May 1993.

[47] Richard Johnson, Jr., Philip Schniter, Thomas J. Endres, James D. Behm, Donald R. Brown, and Raúl A. Casas, "Blind equalisation using the constant modulus criterion: a review," *IEEE Proc. Special issue on blind system identification and estimation*, vol. 86, no. 10, pp. 1927–1949, Oct. 1998.

[48] E. E. Azzouz and A. K. Nandi, "Automatic Identification of Digital Modulations," *Signal Processing*, vol. 47, no. 1, pp. 55 - 69, Nov. 1995.

[49] R. J. A. Little and D. B. Rubin, *Statistical analysis with missing data*, Wiley, 1987.

[50] P. D. Allision, *Missing Data*, Sage, 2002.

[51] Alan Stuart and J. Keith Ord, *Kendall's Advanced Theory of Statistics*, vol. 1, Charles Griffin & Co. Ltd., 5th edition, 1986.

[52] Mohamed Ibnkahla, "Applications of neural networks to digital communications - a survey.," *Signal Processing*, vol. 80, pp. 1185–1215, 2000.

[53] A. K. Nandi and E. E. Azzouz, "Modulation Recognition Using Artificial Neural Networks," *Signal Processing*, vol. 56, no. 2, pp. 165–175, 1997.

[54] M. T. Hagan and M Menhaj, "Training feedforward networks with the marquardt algorithm," *IEEE Trans. Neural Networks*, vol. 5, no. 6, pp. 989–993, 1994.

[55] Martin Riedmiller and Heinrich Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," in *Proc. IEEE Int. Conf. on Neural Networks*, San Francisco, CA, 1993, pp. 586–591.

[56] S. A. Billings and G. L. Zheng, "Qualitative validation of radial basis function networks," *Mechanical Systems and Signal Processing*, vol. 13, no. 2, pp. 335–349, 1999.

[57] L. B. Jack, A. K. Nandi, and A. C. McCormick, "Diagnosis of rolling element bearing faults using radial basis functions," *Applied Signal Processing*, vol. 6, no. 25, pp. 25–32, 1999.

[58] Carl G. Looney, *Pattern recognition using neural networks*, Oxford University Press, 1997.

[59] D. F. Specht, "Generation of polynomial discriminant functions method of pattern recognition," *IEEE Trans. Electronic Computers*, vol. EC-16, pp. 308–319, 1967.

[60] D. F. Specht, "Vectorcardiographic diagnosis using the polynomial dicriminant method of pattern recognition," *IEEE Trans. Biomedical Engineering*, vol. BME-14, pp. 90–95, 1967.

[61] F. Mo and W. Kinsner, "Probabilistic neural networks for power line fault classification," in *Proc. CCECE'98*, Piscataway, NJ, 1998, pp. 585–588, IEEE.

[62] L. A. Branagan and P. D. Wasserman, "Introductory use of probabilistic neural networks for spike detection from an on-line vibration diagnostic system," in *Proc. ANNIE'92*, Fairfield, NJ, USA, 1992, vol. 2, pp. 719–724, ASME.

[63] Donald F. Specht, "Probabilistic neural networks," *Neural Networks*, vol. 3, pp. 109–118, 1990.

[64] P. D. Wasserman, *Advanced Methods in Neural Computing*, chapter 3, pp. 35–55, Van Nostrand Rheinhold, New York, 1993.

[65] Anthony Zaknich, "Introduction to the modified probabilistic neural network for general signal processing applications," *IEEE Trans. Signal Processing*, vol. 46, no. 7, pp. 1980–1990, July 1998.

[66] P. P. Raghu and B. Yegnanarayana, "Supervised texture classification using a probabilistic neural network and constraint satisfaction model," *IEEE Trans. Neural Networks*, vol. 9, no. 3, pp. 516–522, May 1998.

[67] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Stat.*, vol. 33, pp. 1065–1076, 1962.

[68] Simon Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice Hall, 1999.

[69] Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*, John Wiley & Sons, Inc, New York, 2001.

[70] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, 1989.

[71] G. Cybenko, "Approximations by superpositions of sigmoidal functions," *Math. Control, Signals, Systems*, vol. 2, pp. 303–314, 1989.

[72] Pier Luca Lanzi, "Fast feature selection with genetic algorithms: A filter approach," in *Proc. IEEE Int. Conf. Evolutionary Computation*, 1997, pp. 537–540.

[73] Shigeo Abe, Ruck Thawonmas, and Yoshiki Kobayashi, "Feature selection by analyzing class regions approximated by ellipsoids," *IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, vol. 28, no. 2, pp. 282–287, May 1998.

[74] Ron Kohavi and Dan Sommerfield, "Feature Subset Selection Using the Wrapper Method: Overfitting and Dynamic Search Space Topology," in *Proc. Int. Conf. Knowledge Discovery and Data Mining (KDD-95)*, August 1995, pp. 192–197.

[75] Ron Kohavi and George H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–323, 1997.

[76] Jihoon Yang and Vasant Honavar, "Feature subset selection using a genetic algorithm," *IEEE Trans. Intell. Syst.*, vol. 13, no. 2, pp. 44–49, March/April 1998.

[77] L. B. Jack and A. K. Nandi, "Genetic algorithms for feature selection in machine condition monitoring with vibration signals," *IEE Proc. Vision, Image and Signal Processing*, vol. 147, no. 3, pp. 205–212, 2000.

[78] Lindsay Jack, *Applications of Artificial Intelligence in Machine Condition Monitoring*, Ph.D. thesis, The University of Liverpool, 2000.

[79] Mineichi Kudo and Jack Sklansky, "Comparison of algorithms that select features for pattern classifiers," *Pattern Recognition*, vol. 33, pp. 25–41, 2000.

[80] Melanie Mitchell, *An Introduction to Genetic Algorithms*, MIT Press, Cambridge, MA, 1996.

[81] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Mag.*, vol. 2, pp. 559–572, 1901.

[82] Ian T. Jolliffe, *Principal Component Analysis*, Springer-Verlag, New York, 1986.

[83] P. F. Baldi and K Hornik, "Learning in linear neural networks: A survey," *IEEE Trans. on Neural Networks*, vol. 6, no. 4, pp. 837–858, 1995.

[84] E. Oja, J. Karhunen, L. Wang, and R. Vigario, "Principal and independent components in neural networks recent developments," in *Proc. VIIth Workshop on Neural Nets*, 1995.

[85] K. I. Diamantaras and S. Y. Kung, *Pricipal Component Neural Networks: Theory and Applications*, Wiley-Interscience, New York, 1996.

[86] Pierre Comon, "Independent component analysis, a new concept?," *Signal Processing*, vol. 36, no. 3, pp. 287–314, Apr 1994, Special issue on Higher-order statistics.

[87] Christian Jutten and Jeanny Herault, "Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, no. 1, pp. 1–10, Jul 1991.

[88] M. Gaeta and J. L. Lacoume, "Sources separation without a priori knowledge: the maximum likelihood solution," in *Proceedings of the EUSIPCO'90*, Enrique Masgrau Luis Torres and Miguel A. Lanunas, Eds., Barcelona, Spain, Sep. 1990, vol. 1, pp. 621–624, Fifth european signal processing conference, Sept. 18-21 1990.

[89] Aapo Hyvärinen, "Fast and robust fixed-point algorithms for independent component analysis," *IEEE Trans. Neural Networks*, vol. NN-10, no. 3, pp. 626–634, May 1999.

[90] H. B. Barlow, "Possible principles underlying the transformations of sensory image," *Sensory Communication*, pp. 217–234, 1961.

[91] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, pp. 607–609, 1996.

[92] A. D. Parkins and A. K. Nandi, "Simplifying hand written digit recognition using a genetic algorithm," in *XI European Signal Processing Conf.*, Toulouse, France, 2002, vol. 1, pp. 111–114.

[93] Liang Hong and K. C. Ho, "An antenna array likelihood modulation classifier for BPSK and QPSK signals," in *Proc. MILCOM'01: Communications*

for Network Centric Operations: Creating the information force, 2001, pp. 118–112.

[94] C. Huang and A. Polydoros, "Likelihood methods for MPSK modulation classification," *IEEE Trans. Commun.*, vol. 43, pp. 1493–1504, Feb/Mar/April 1995.

[95] Andreas Polydoros and Kiseon Kim, "On the detection and classification of quadrature digital modulations in broad-band noise," *IEEE Trans. on Commun.*, vol. 38, no. 8, pp. 1199–1211, August 1990.

[96] Vicente Zarzoso, *Closed-form higher-order estimators for blind separation of independent source signals in instantaneous linear mixtures*, Ph.D. thesis, The University of Liverpool, 1999.

[97] Frank Herrmann, *Independent Component Analysis with Applications to Blind Source Separation*, Ph.D. thesis, The University of Liverpool, 2000.

[98] Franck Harroy and Jean-Luis Lacoume, "Maximum-likelihood estimators and Cramér-Rao bounds in source separation," *Signal Processing*, vol. 55, pp. 167–177, 1996.

[99] Vicente Zarzoso and Asoke Kumar Nandi, "Blind separation of independent sources for virtually any source probability density function," *IEEE Trans. Signal Processing*, vol. SP-47, no. 9, pp. 2419–2432, Sep. 1999.

[100] Nirode Mohanty, *Random Signals Estimation and Identification: Analysis and Applications*, Electrical/ Computer Science and Engineering Series. Van Nostrand Reinhold Company, 1986.

[101] Free Software Foundation, "Gnu radio: The gnu software radio," Internet, http://www.gnu.org/software/gnuradio/gnuradio.html.

[102] Robert H. Morelos-Zaragoza, "Joint phase-lock detection and identification of digital m-psk/m-qam modulation," in *Proc. 2000 Third Generation Wireless Communication Conference*, San Francisco, June 2000, pp. 272–279.

[103] R. H. Morelos-Zaragoza and Kenta Umebayashi, "Adaptive carrier recovery with modulation identification," in *Proc. 2001 Int. Symp. on Signals, Systems, and Electronics*, Tokyo, July 2001, pp. 216–219.

[104] Ashit Talukder and David Casasent, "A closed-form neural network for discriminatory feature extraction from high-dimensional data," *Neural Networks*, vol. 14, pp. 1201–1218, 2001.

[105] Aapo Hyvärinen, Juna Karhunen, and Erkki Oja, *Independent Component Analysis*, Wiley-InterScience, 2001.