Tracking changes in transmitter modulation type in a non-cooperative environment

LY Uys

orcid.org/0000-0001-5268-1053

Dissertation submitted in fulfilment of the requirements for the degree *Master of Engineering* in Computer and Electronic Engineering at the North-West University

Supervisor: Prof ASJ Helberg

External co-supervisor: JJ Strydom

Graduation May 2018

Student number: 23509775

ABSTRACT

Automatic modulation classification (AMC) is a challenging task in a non-cooperative environment where channel state information and signal parameters are not always available. Non-cooperative transmissions in military environments may be hampering or threatening to a user's own goals. In this environment signals can use never before seen modulation types or even modulation types that are specifically designed to avoid interception, detection and classification. Modulation is in effect used here as another layer of encryption. Modulation types thus have to be classified blindly, that is, without the use of a priori signal and channel state information. Adaptive modulation techniques complicate the task of classifying adversaries' signals even more. It is desirable to be able to track the changes in an adversary emitter's modulation type, because the transmitter may be identified or their messages may be recovered, which is a critical aid in supporting battlefield decision making.

The objective of this study is to classify and track changes of modulation types from a communications transmitter in a non-cooperative environment without channel state information. The secondary objective is to develop the method in such a way that the digital signal processing components thereof can be implemented on a hardware platform provided by the CSIR.

Communication signals with modulation types Amplitude Shifts Keying (ASK) of order two and four, Phase Shift Keying (PSK) of order two and four, and Frequency Shift Keying (FSK) of order two and four were considered. The channel effects that were considered were AWGN noise and flat fading in a static multipath Rayleigh fading channel.

A literature study was first performed to identify candidate algorithms for AMC that can be implemented on a hardware platform and the best classification algorithm that met the research objectives was selected. The performance of the selected algorithm was evaluated in both software and hardware under varying channel conditions whereafter the results were analysed and compared. The tracking of changes from one modulation type to another was performed by logging the modulation type over time.

Feature based classification was selected to classify and track modulation types of a signal. Features based on the instantaneous amplitude, phase and frequency of a signal were used for feature extraction and a decision tree was used for classification. The method was tested under varying SNR conditions from 0 dB to 30 dB in an AWGN channel and flat fading conditions in a multipath Rayleigh fading channel at an SNR of 30 dB and 10 dB. Classification accuracy higher than 99 % was achieved on average for the SNR conditions. Classification performance of 97% and 93% was achieved on average for the fading conditions at 30 dB and 10 dB SNR respectively in software. The classification performance for hardware was 89% and 71% on average for the fading conditions at an SNR of 30 dB and 10 dB respectively. It was found that signal length has a significant effect on the classification performance.

Keywords- Automatic Modulation Classification, Feature based classification, non-cooperative environment, Rayleigh flat fading, decision tree

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Prof Albert Helberg, for his support and guidance during the course of this study. His knowledge, passion and love for his work have inspired me. I would also like to thank the North-West University for the opportunity to perform this study.

I would also like to thank Jurgens Strydom, my external co-supervisor and mentor, for his guidance and council in both my professional work and personal life. He has helped me grow in many aspects of my life. I would also like to thank Marcel Gouws for his valuable inputs during the first part of this study.

I would like to thank my family and friends for their continual support. Your love and encouragement carried me through these two years.

I would like to thank the CSIR for their financial contribution to complete this study. I would like to thank the Radar and Electronic Warfare group within the DPSS unit of the CSIR for providing required resources for the completion of this study.

Most importantly, I would like to thank my Lord, Jesus Christ, for His faithfulness and grace upon my life. I would like to thank my heavenly Father for blessing me with the ability to complete this study.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Background

The utilisation of the radio frequency (RF) spectrum includes communications, radio navigation, television- and radio broadcasting, and remote sensing of objects, areas or phenomena [1]. The utilisation of the RF spectrum has witnessed a great increase in the last few decades and continues to do so. Increases in numbers of wireless devices, technologies and applications as well as the constant drive towards higher data rates has led to congestion of the RF spectrum.

Spectral congestion has led to the development of new technologies in military and civilian applications. Traditionally systems used to depend on fixed modulation and spectrum allocation. These systems are however being replaced by more advanced systems that are spectrum aware and able to adapt to the environment or situation. The systems change their parameters over time, which result in dynamic behaviour. The systems have capabilities such as frequency- and modulation agility which are used to compensate for the scarcity of available frequency bands. Techniques such as Automatic modulation classification (AMC) are required to automatically identify the modulation type of signals in order for receivers to select the correct demodulation method.

Cooperative transmissions are a communication system's own transmissions that are under the control of the transmitter receiver pair and are used to achieve the transmitter and receiver's goals in the RF spectrum. Non-cooperative transmissions are transmissions that are not under a transmitter and receiver pair's control. In the non-cooperative scenario, channel state information (CSI) and signal parameters may be unknown to the receiver and may be hampering or threatening to a transmitter and receiver pair's own goals. From the perspective of this study, non-cooperative transmissions typically represent either illegal civilian transmissions or transmissions from adversaries in military scenarios. Thus the non-cooperative nature of some signals necessitates the requirement for AMC with no channel state information available, also known as blind AMC.

1.1.1 Automatic Modulation Classification

Modulation classification was initially done manually by signal engineers who were trained to identify several signal formats [2]. One of the most common methods still used today for modulation classification, as described in [3] and [4], is the use of a computer-based library that contains knowledge of known signal parameters gathered previously through electronic intelligence operations. Human signal engineers classify these recorded signals offline with the assistance of computer methods and add them to the computer based library.

Systems that classify signals based upon a fixed library containing a pre-determined set of emitter characteristics have become unable to handle transmissions from dynamic emitters. The classifiers are only able to classify a fixed variety of signals and their adaptation capability to new and unknown signals is limited. New and unknown signals need to be recorded and analysed in a laboratory, taking multiple days or hours. The systems are then retrained with knowledge of the previously unknown signals and redeployed in the field. Such processes are too slow and place military forces at a disadvantage as new unknown signals may appear in the field by the time systems are retrained [5]. The manual process of modulation classification was later automated with automatic modulation classifiers which contributes to reducing the time taken to classify systems, and especially helps in the case where emitters are dynamic [2], [6].

AMC is used to automatically ascertain the modulation type of a signal by applying one or more signal processing techniques and classification algorithms to the signals sensed in the environment [7]. It is often referred to as "an intermediate operation between signal detection and demodulation or system reaction" [8]. In the military domain, the AMC technique is critical for the purpose of electronic warfare.

1.1.2 AMC in Electronic Warfare

Electronic Warfare (EW) is any action that involves the use of electromagnetic or directed energy by military forces to attack an adversary, to control the utilisation of the EM spectrum, and to protect systems against attacks. EW exploits the electromagnetic (EM) spectrum by sensing, intercepting, manipulating, hardening and analysing signals to determine enemies' applications of the spectrum and enforces suitable measures with the aim of control of the spectrum when necessary [9].

EW includes three top level operational functions: electronic attack (EA), electronic protection (EP) and electronic support (ES) [9]. EA uses EM energy to attack electronic facilities and equipment with the purpose of degradation, neutralisation or destruction of enemy combat capability. It includes actions such as jamming, which is the primary measure for the prevention of communication between adversaries, and deception [10]. EP includes actions to protect the host platform from either friendly or hostile EW employment with the purpose of degradation, neutralisation or destruction of friendly combat capability. ES searches for and intercepts intentional or unintentional EM emissions to record, analyse, locate, and identify them in order to allow effective decision making for military operations. For a complete EW capability these three functions of EW are closely interconnected [9].

The need for automatic modulation classification (AMC), according to [2], first arose in military scenarios where modulation classification is required in Electronic Warfare (EW) systems for identification of adversary emitters, preparation of jamming signals and recovery of intercepted signals. The use of a modulation classifier in EW systems is illustrated in Figure 1.

AMC is important for all three top level functions of EW. The knowledge of the modulation type can be used in ES to determine the appropriate demodulation method for intercepted signals. Messages transmitted from adversaries can then be recovered with the help of signal decrypting- and translating processes. AMC can also assist ES in classification, identification and the locating of adversary units. AMC can assist in determining the appropriate jamming technique in EA by identifying the modulation type and altering the jammer to modulation changes. The two most common jamming techniques are the emission of noise and spoofing. Spoofing includes the emission of false signals with the same modulation type and frequency as an adversary signal. As already mentioned, the goal of EP is to protect the military force's own systems from an adversary's EA measures. The military force's own systems can be prevented from being jammed by monitoring the modulation type of the jamming signal and changing the modulation type of its own signals to make it more robust against the adversarial signal [10].

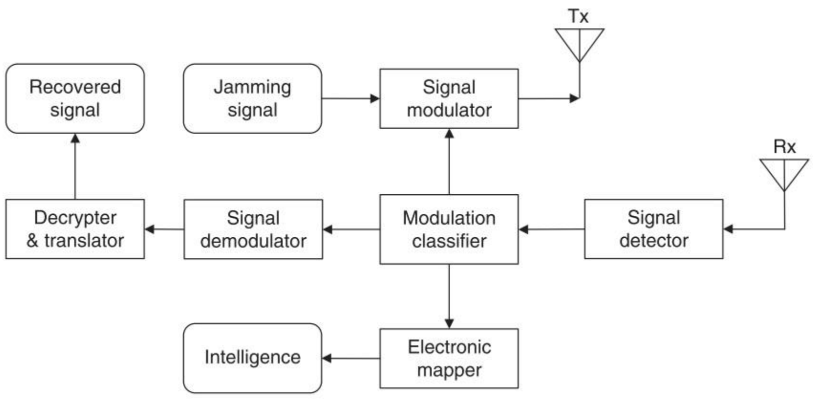


Figure 1: Military Signal Intelligence System [10]

AMC also has applications in civilian scenarios such as identifying interference sources, monitoring spectrum activities, detecting unlicensed users and managing the spectrum [11], [12]. AMC is a critical component of dynamic spectrum access/management (DSA) in the context of cognitive radios. AMC is used to sense and detect the absence or presence of primary users (PU) who have licenses for allocated frequency bands in the spectrum [12], [13]. Cognitive radios (CR), also known as secondary users (SU), then through the use of CR techniques intelligently access vacant channels while avoiding channels that are occupied by primary users (PU) [12]. For a SU to successfully operate in the DSA context, it needs to track modulation changes over time to ensure it continues to allow unaffected access for PUs.

1.1.3 AMC for Modulation Change Tracking

With the ever growing increase in utilisation of the spectrum, there are still challenges with regards to AMC that need to be addressed including tracking of transmitter modulation changes, i.e. logging the modulation type over time, through blind modulation classification.

The tracking of modulation changes has been investigated in applications such as use of link adaption (LA), also known as adaptive coding and modulation (ACM) [10], [14]. Link adaption is where a single transmitter can employ multiple modulation types to control the data rate and bandwidth usage, in an effort to guarantee the integrity of the message. A modulation type is selected from a pool of candidate modulations according to channel conditions and system specifications. The receiver has to know the modulation type in order to demodulate the received signal successfully. Information on the modulation type can be included in the transmitted signal to notify the receiver about modulation changes; however the spectrum efficiency is reduced by this method due to the additional modulation information overhead required. To overcome this problem, the modulation type of the received signal can be automatically identified through blind AMC [10].

Another application of modulation change tracking occurs in AMC for adaptive power control in cognitive communications which is an interference avoidance technique in civilian cognitive radio applications [15], [16]. A PU's allocated frequency band is accessed by a SU based on an Adaptive Coding and Modulation (ACM) protocol. Once the modulation type of the PU is identified, a power control scheme is used by the SU. The SU attempts to access the PU's band and, if successful, increases its transmitting power until the PU changes its modulation type on the assumption that the modulation change is due to the interference caused by the SU. As soon as the change in modulation type of the PU is detected, the SU reduces its transmitting power in an attempt to control the induced interference and allow both the PU and SU to utilise the channel [15].

Modulation change tracking is also used in DSA applications [17], [18]. The transmitter changes the modulation type according to the channel conditions and level of interference when occupying different available bands in the spectrum, known as white spaces, with different operating frequencies. The receiver has to constantly monitor the modulation type used by the transmitter for correct demodulation of the received signals [17].

The applications discussed above occur in cooperative environments. AMC is however a challenging task in non-cooperative environments. In both military and civilian spectrum use cases, the spectrum can contain signals from cooperative and non-cooperative communication systems. In a cooperative environment, a pool of candidate modulation types and a priori knowledge can be utilised to perform modulation classification, greatly simplifying the task. For unknown signals found in a military environment a pool of candidate modulation types is not always available or accurate enough to assist the classifier. There may even be never before seen modulation types as well as modulation types that are designed to avoid interception, detection and classification. Modulation is in effect used here as another layer of encryption to prevent adversaries from recovering their messages [19].

An example where adaptive modulation techniques are used to obscure transmissions is found in [20]. The paper discusses case studies of attacks targeting tactical military software defined radios (SDRs) in which adversaries identify vulnerabilities in the radio sets or in the communication channel between radio sets. The authors recommend the use of adaptive modulation techniques for transmission security in future development of new systems and architectures.

The only study found on the topic of the tracking of changes in modulation types in a non-cooperative environment was [21]. This study proposed a method for the detection of cognitive radios that use the spectrum illegally. These CRs avoid being charged for the use of the RF spectrum by hiding themselves between PUs. Changes in their signal parameters, such as modulation types, are tracked and the CRs are then detected accordingly.

The challenge in our study is therefore to investigate the tracking of modulation changes in communications signals in a non-cooperative environment, specifically in military scenarios where frequently changing adaptive modulation types are used by adversaries to contribute in obscuring their transmissions.

The algorithms that are already developed for the tracking of changes in modulation types have been developed for signals in cooperative environments where assistance and a priori information about signal parameters are available. These algorithms will not necessarily be suitable for utilisation in non-cooperative environments and the classification accuracy may be inferior, which is a vital factor in military applications when suitable measures against adversaries need to be taken.

1.2 Problem Statement

In a cooperative environment, a pool of candidate modulation types and a priori knowledge can be utilised to perform automatic modulation classification. For unknown signals found in a military environment a pool of candidate modulation types is not always available or accurate enough to assist the classifier. There may even be never before seen modulation types as well as modulation types that are designed to avoid interception, detection and classification. This is a problem because the modulation type of a signal effectively provides another layer of encryption in non-cooperative environments where a priori signal and channel state information are unknown. Signal parameters first have to be estimated and channel state information has to be determined for accurate classification. Adaptive modulation techniques complicate the task of classifying adversaries' signals even more, because the signal modulation type changes quickly with time. It is desirable to be able to track the changes in adversary emitters' modulation type. When the change from one modulation type to another modulation type of signals from a transmitter can be tracked, the transmitter may be identified or their messages may be recovered which is a critical aid in supporting battlefield decision making.

The classification of the modulation type has to occur as quickly as possible in order to keep up with the change from one modulation type to another performed by the transmitter. The speed and processing power of a system required to process data for classification is thus important. The classifier also needs to be capable of classifying a wide range of modulation types in order to be able to keep tracking the varying modulation types.

1.3 Research Objective

The objective of this study is to classify and track changes of modulation types from a communications transmitter in a non-cooperative environment without channel state information. The secondary objective is to develop the method in such a way that the digital signal processing components thereof can be implemented on a hardware platform provided by the Council for Scientific and Industrial Research (CSIR).

A complete capability required to track changes in transmitter modulation types includes the ability to receive and digitise signals of interest, spectrum sensing functionality to detect signals of interest, signal parameter estimation, classification of signal modulation type, and the tracking of changes in that modulation type. This study focuses on the latter two steps, namely on developing a method capable of tracking changes in modulation types through classification of signal modulation type without the use of channel state information. This study focuses on the classification of communication signals, more specifically signals with digital modulation types of Amplitude Shift Keying (ASK) of order two and four, Phase Shift Keying (PSK) of order two and four, and Frequency Shift Keying (FSK) of order two and four. Modern communication systems make more use of digital signals instead of analogue signals. The main reason for this is that digital modulations are better suited to digital data and are more robust against interference. The focus of this study is on a larger number digital modulation types with lower orders rather than fewer modulation types that included higher orders. This approach is chosen to create a baseline on which future work could expand to include higher order modulation types.

A digital radio frequency memory (DRFM), which is used for EW operations, is the target platform for hardware implementation [22]. The following elements of this system were provided for creation of the hardware demonstrator and are not developed within the scope of this study:

- RF hardware
- Digital hardware
- Digital signal front-end processing firmware
- Existing auxiliary firmware interfaces and modules
- Test software

1.4 Research Methodology

In order to accomplish the research objective discussed above, the following methodology was followed. The first step is to identify candidate algorithms for AMC that can be implemented on a hardware platform. A literature study is performed to identify the state of the art in this field. The literature is critically evaluated and the best classification algorithm that meets the research objectives is selected. The selected algorithm is then evaluated in detail through simulation, whereafter a subset of the algorithm is implemented on a hardware platform. The performance of the selected algorithm is evaluated in both software and hardware in varying channel conditions, namely white noise and static flat Rayleigh fading, whereafter the results are analysed and compared. The outcome of the study is compared with the research objective, and critically evaluated in that context.

The signal models used for the simulated signals are selected such that they create meaningful scenarios to evaluate the performance of the algorithm and provide credible test data for real world applications. The signal models include noise models and static flat Rayleigh fading channel models which set limitations for accurate classification.

AMC and tracking is simulated and tested in software and the algorithm for hardware implementation is developed and implemented on a concept demonstrator. The concept demonstrator is also tested with simulated data satisfying the same criteria mentioned for the software simulation. The results of the hardware implementation are compared to the software simulation results to show the validity of the hardware implementation.

1.5 Structure of Dissertation

This thesis documents the research outlined in this chapter as discussed in the background, problem statement, research objective and research methodology. The structure of the thesis closely follows the approach outlined in the research methodology.

Chapter 2 consists of a literature study of the aspects that need to be considered to perform AMC. A signal model describing the signal parameters and channel effects is derived. A study on modulation classification is performed to obtain a suitable technique for AMC with regards to the research objectives of this study. The identification of different machine learning techniques as well as feature selection for machine learning techniques is performed.

Chapter 3 presents the conceptual design which describes the process followed for the development of the AMC algorithm for both software simulation and hardware implementation with the aid of functional flow diagrams.

Chapter 4 documents the simulation of communication signals for both the software and hardware implementation of the method, using the signal models derived in Chapter 2. The effect of signal processing and representation thereof are also derived. The simulation of the AMC algorithm in software is described as well as the hardware implementation. Datasets are generated for both approaches and results are obtained.

Chapter 5 concludes the work by discussing the performance of the algorithm with regards to the channel effects. The findings of the work are discussed and future work is identified.

2 LITERATURE STUDY

In order to select a suitable AMC algorithm for blind classification operable in real world scenarios, the algorithm must be applied to realistic signals. A model, representing a signal propagating through a channel with real world effects, will thus be derived first. The selected AMC algorithm has to be able to operate under the channel conditions selected. After the signal model is derived, there are several other metrics to take into consideration when comparing and selecting a suitable AMC algorithm. The system requirements determine the priority of the metric. The AMC algorithm meeting the requirements can then be selected. Techniques optimising the AMC algorithm, also meeting the requirements, will then be discussed.

2.1 Signal Model

The description of the signal model includes the transmitted signal, the effects of the channel on the signal propagating through the channel and finally the received signal.

2.1.1 Digital Transmitted Signal

Modern communication systems make more use of digital signals instead of analogue signals. The main reason for this is that digital modulations match digital data better and are more robust against interference [23]. The transmitted signal for digitally modulated signals can be presented by:

$$s(t) = A(t)\cos\big(2\pi f_c t + \phi(t)\big) \tag{1}$$

$$= Re\{\tilde{s}(t)e^{j2\pi f_c t}\} \tag{2}$$

where $A(t)$ is the amplitude, f_c is the carrier frequency, $\phi(t)$ the phase of the signal and $\tilde{s}(t) = A(t)e^{j\phi(t)}$ represents the complex baseband signal [24].

2.1.2 Channel Parameters and Effects

2.1.2.1 Additive White Gaussian Noise

One of the most widely used noise models for communication channels is the Additive White Gaussian Noise (AWGN) model [24]. Wideband Gaussian noise is caused by thermal vibrations in conductors as well as radiation from several sources. Over the bandwidth of interest, the Gaussian noise is assumed to be flat and white, which means that the noise samples are uncorrelated [24]. The probability density function of the Gaussian distribution is given by:

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3}$$

where μ and σ^2 are the mean and variance of the distribution respectively [23].

In the AWGN model, noise with Gaussian distribution and zero-mean is added to the signal. The AWGN model is the elementary limitation on the accuracy of modulation classification and is used in most literature on modulation classification [23].

2.1.2.2 Fading

There are various phenomena in a wireless communication channel which alter a signal as it propagates through the channel. One of the primary effects is fading [25]. Fading is defined as "the variation in signal amplitude at the receiver caused by the characteristics of the signal path and changes in it" [25]. The effects can be categorised as large-scale effects and small-scale effects [24]. Large-scale effects cause slow fading and shadow fading due to the properties of the general terrain. When large objects such as buildings and hills are present, signals are not prevented from being propagated, but diffraction allows signals to propagate around the objects at a reduced power level. These effects change relatively slowly with time and they are taken into consideration with the prediction of coverage and service availability [24].

The small-scale effects change much faster than the large-scale effects relative to the properties of a transmitted signal. Small-scale effects are taken into consideration with the design of transmitters and receivers as well as the selection of modulation types to be used [24]. Small-scale effects cause fast Rayleigh fading due to the local environment and movement in the channel within that environment [24]. Reflections against trees and buildings may cause a transmitted signal to arrive at the receiver over multiple different paths and at different time instants causing multiple signals to arrive at the receiver each with its own amplitude, phase and time delay. This is known as multipath propagation. Because all of these signal components add up at the receiver, they may interfere with each other destructively or constructively. If there is motion in the channel, an additional effect caused by how the multiple paths vary over time, is present. This second effect causes distortion due to the Doppler shift [24]. The two types of effects can be described by the delay spread and the Doppler spread of the channel [26].

The multipath delay is described by the delay spread. The delay spread is the second central moment of the power delay profile (PDP) [10]. The PDP gives an estimation of the average power in the multipath and can be seen in Figure 2. First the average delay is calculated by:

$$\bar{\tau} = \frac{\sum P_h(\tau)\tau}{\sum P_h(\tau)} \tag{4}$$

Where τ and $P_h(\tau)$ are the delay and power of the individual paths.

The average delay spread is defined as:

$$\overline{\tau^2} = \frac{\sum P_h(\tau)\tau^2}{\sum P_h(\tau)} \tag{5}$$

The RMS delay spread is given by:

$$\sigma_T = \sqrt{\overline{\tau^2} - \bar{\tau}^2} \tag{6}$$

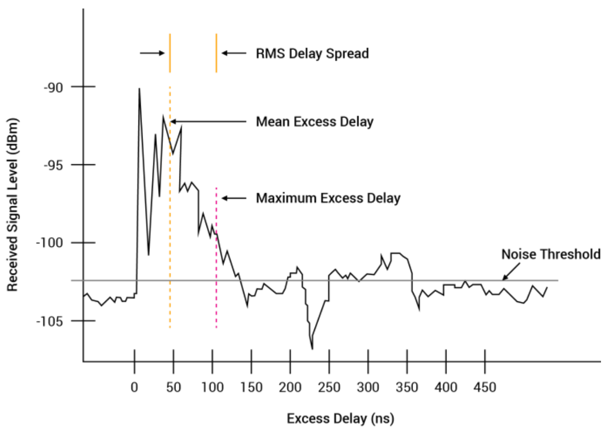


Figure 2: Power Delay Profile [27]

The Delay spread causes two types of fading: frequency-flat fading and frequency selective fading. Frequency flat fading occurs when the symbol time is greater than the delay spread or equivalently when the signal bandwidth is smaller than the coherence bandwidth. The coherence bandwidth can be defined as bandwidth over which the frequency correlation is strong [26]. This is the bandwidth over which all the frequency components are passed by the channel with nearly equal gain and linear phase. The signal experiences constant attenuation and phase shift over the transmission period. In contrast, frequency selective fading occurs when the symbol time is less than the delay spread or equivalently when the signal bandwidth is greater than the coherence bandwidth. This effect results in the introduction of inter symbol interference by the channel [24].

The movement of the receiver, transmitter or any other objects within in a channel, from which signals may reflect, introduces changes in the signal frequency. This is known as the Doppler Effect [25]. The Doppler Effect causes two types of fading: time-flat fading and time-selective fading also known as slow fading and fast fading respectively. Slow fading is a large scale effect caused by reflections of signals from large objects that are far from the transmitter or receiver [24]. The movement in the channel is slow relative to the objects. The changes in the frequency are therefore small and the symbol time is smaller than the coherence time of the channel. The coherence time can be defined as the period over which the correlation of the channel impulse response is strong [26]. The channel is thus almost constant over at least one symbol duration. Fast fading occurs when there are large changes in the signal frequency due to the movement in the channel [25]. The movement is fast relative to local objects in the environment [24]. The symbol time is larger than the coherence time of the channel. The impulse response changes rapidly within the symbol duration of the signal which leads to distortion due to frequency dispersion [26].

Channels can thus be classified into one or more of the following types: Time-flat, time-selective, frequency-flat and frequency selective. Channels are classified based on the signal to be transmitted and carried through that channel. Narrowband signals in mobile channels often experience flat-fading, i.e. flat-frequency and time-selective fading [24]. For most radio channels with transmission frequencies less than 1 GHz the coherence bandwidths are normally tens of kilohertz. High frequency (HF) radio channels are however an exception, where narrow band channels can be frequency selective due to propagation modes. Wideband channels are often both frequency selective and time selective, when either the transmitter or receiver is in motion [24].

Narrowband static channels are considered in this study. In these channels, multipath interference (Rayleigh fading) and shadow fading occur the most [25]. The focus is therefore on the effect of the delay spread due to multipath propagation in a static Rayleigh fading channel.

2.1.3 Received Signal with Channel Effects

A frequency-flat Rayleigh fading channel is modelled as a linear filter with an impulse response given by:

$$h(t,\tau) = \sum_{i=0}^{L-1} \tilde{\alpha}_i \delta(\tau - \tau_i) \tag{7}$$

where L is the number of multipaths, $\tilde{\alpha} = \alpha_i e^{j\theta_i}$ is the i^{th} path complex gain and τ_i the i^{th} path delay. The complex gain is assumed to be constant in a static channel [24].

In a multipath Rayleigh fading model, the phases of the various path components are independent and uniformly distributed between [0, 2π] and the real and imaginary components of the complex gain of each path are zero mean Gaussian random variables that are independent and identically distributed (i.i.d) [24].

The received passband signal is the sum of the various multipath components after the signal has propagated through the fading channel and is given by:

$$r_p(t) = \sum_{i=0}^{L-1} Re\left\{\alpha_i e^{j\theta_i} \tilde{s}(t - \tau_i) e^{j2\pi f_c t}\right\} + n(t) \tag{8}$$

where $n(t)$ is the additive white Gaussian noise [24].

2.2 Automatic Modulation Classification

Automatic Modulation Classification (AMC) is used to automatically ascertain the modulation type of a signal, by applying one or more signal processing techniques and classification algorithms to signals sensed from the environment [28]. AMC is used for a wide variety of RF spectrum applications including multiple signal classification [29], [30], [31]; classification in multipath fading channels [32], [33], [34]; dynamic spectrum access [17], [18]; blind modulation classification [35], [36], [37], [38], [39]; classification of orthogonal frequency-division multiplexing (OFDM) signals [40], [14], [41] and link adaption [10], [14], [42], [43], [44], [45], [46], [47].

There are two general approaches for the AMC of signals: likelihood-based (LB) classification and feature-based (FB) classification [48]. LB classification formulates the classification as a composite hypothesis-testing problem which assigns each candidate modulation type to the incoming signal under the hypothesis H_i. The likelihood function is then used to find the correct modulation type of the signal. FB classification entails 2 steps, feature extraction and decision making. For feature extraction, a carefully selected set of hand crafted features are extracted from the signal of interest. A decision (classification of modulation type) is made based on the values of the features.

2.2.1 Likelihood Based Classifiers

Likelihood based classifiers minimises the probability of incorrect classification [49]. When channel state information is known, LB classification is an optimal approach for AMC [49]. LB classifiers are able to classify digital modulation types including M-ASK, M-PSK, M-FSK, M-PAM, M-QAM [49], [50] and [51]. From surveys on AMC in [49], [51] and [50] four general likelihood based classifiers have been identified. They include Maximum likelihood (ML) [51], [52], [53], average likelihood ratio test (ALRT) [49], [51], [54], [55], [56], General likelihood ratio test (GLRT) [49], [51], [57], [58], [59] and Hybrid likelihood ratio test (HLRT) [49], [50], [59], [60], [61].

For a maximum likelihood classifier, the likelihood for each modulation hypothesis is tested. The likelihoods of the different hypotheses are compared and the maximum likelihood among all the candidate likelihoods is selected as the classified modulation type. With perfect channel knowledge, the ML method has very high classification accuracy because the computations are repeated for each modulation hypothesis. Furthermore, all the channel parameters must be known [51]. This method is also not robust against phase and frequency offsets [51] and it is more likely to classify a signal as a certain modulation type with denser I-Q constellations [42].

The next method, ALRT, treats unknown channel parameters as random variables and the likelihood function is calculated by taking the average over these variables. Each unknown parameter is replaced with an integral which includes all possible values of the unknown parameter and its corresponding probabilities [50]. The integration operations make this method more computationally complex and with many unknown parameters, this method becomes impractical [49].

GLRT is a combination of maximum likelihood estimation and classification [51]. An unknown parameter is estimated under the assumption that the hypothesis H_i is true. The maximum likelihood estimates over each unknown parameter are then used in the likelihood ratio test [50]. GLRT is less complex than ALRT by avoiding the integration calculations. The noise power also does not have to be known in order to compute the likelihood function of GLRT [49]. It is however a biased classifier towards higher order modulation types [51]. The likelihood for lower- and higher order modulation types are equal when lower order modulation types, e.g. 4-QAM and 16-QAM, are classified [51], [59].

HLRT is a combination of ALRT and GLRT classifiers. The likelihood function is obtained by taking the average over the data symbols of a signal. The resulting likelihood function is then maximised with respect to the unknown parameter and the bias classification problem is in so doing removed [49], [51]. Additionally, HLRT is less computationally complex than ALRT, and achieves better classification performance compared to ALRT and GLRT. It is however more computationally complex than GLRT due to the exponential functions [49]. With several unknown parameters, this method becomes very time consuming when finding the maximum likelihood estimates of the parameters [49], [50]. Other less complex methods for parameter estimation can be used instead which is then what is described as a quasi-HLRT classifier in literature [49], [50], [62].

Expectation maximisation (EM) is used in conjunction with the ML classifier in [63], [53], [64] in the case where multiple unknown channel parameters need to be estimated. EM is an iterative process with two steps: an expectation step and a maximisation step. After initial estimated values are assigned to the unknown parameters, the expectation step evaluates the likelihood of the estimation. The maximisation step aims to maximise the likelihood function of the current iteration. This process is repeated until convergence is reached or a predefined number of iterations are executed, for each modulation hypothesis of the ML classifier [63]. When compared to other classifiers (including an ML-classifier, a distribution-test based classifier, a moments FB classifier and a cumulants FB classifier), the EM-ML classifier showed to have the highest accuracy and proved to be more robust against AWGN and channel conditions [65]. With this method compensation of phase and frequency offsets are possible in the estimation stage of the channel parameters [65]. However, the computational complexity of the EM-ML classifier was the highest among all the classifiers in the complexity comparison due to the iterative estimation process [65].

2.2.2 Feature Based Classifiers

The computational complexity of likelihood classifiers gives rise to suboptimal classifiers with smaller computational cost such as feature based classifiers [49]. If feature based classifiers are properly designed, their performance can be near-optimal [50]. FB classifiers are able to classify digital modulation types including M-ASK, M-PSK, M-FSK, M-PAM, M-QAM [49], [50], [66] and some FB classifiers are able to also classify analogue modulation types including SSB, DSB, AM, FM and VSB [66], [67], [68]. Three main feature based classification methods include the extraction of features based on the instantaneous amplitude, phase and frequency [50], [67], [69], [68], [70], features based on the wavelet transform [50], [71], [72] [73], [74] and features based on higher order statistics of the signal [17], [33], [75], [76], [77], [78], [79].

The first method separates a pool of modulation types into subsets according to the properties contained in the instantaneous amplitude, phase and frequency of the different modulation types. The features based on the instantaneous amplitude, phase and frequency are used sequentially to distinguish between subsets until each modulation type is discriminated. A decision tree is often used for this FB method [66]. FB classification based on the instantaneous information is the most intuitive way to determine the modulation type of a signal [50] and has a simple implementation [17]. This method can also classify a wide variety of analogue and digital modulation types [66], [67], [68]. FB classification based on the instantaneous information however relies on feature value thresholds to be set in advance, which makes it more sensitive to noise and other channel effects [17]. From literature it is also evident that this method is not utilised for classification of modulation types with orders higher than four. Per illustration, [69] shows a case where by choosing a second set of thresholds, modulation types of order eight can also be distinguished. When the number of samples for calculation was increased, the results showed that good classification accuracy can be attained at an SNR of 10 dB.

Wavelet transform based features are used to localise the transients in the instantaneous amplitude, phase and frequency of the received signal. After the wavelet transform is applied to the signal, the transient characteristics are extracted. The differences in transient characteristics of signals are used to distinguish between the different modulation types. This method is more robust against noise than instantaneous based features, but it has higher computational complexity than instantaneous based features [17]. This method has however been implemented on hardware in [80]. A drawback of features based on the wavelet transform can only classify between FSK, PSK and QAM signals. Other feature based methods such as higher-order statistics, are needed to discriminate between QAM and ASK signals [66]. There are instances in literature where classification of other modulation types occur where single carrier signals are distinguished from OFDM signals [81]; and where QPSK signals are distinguished from Gaussian Minimum Shift Keying (GMSK) signals [74]. Only two modulation types can be discerned from each other in these instances.

Higher order statistic features include the calculation of moments and cumulants of signals. These features characterise the shape of the distribution of the I&Q samples of a signal [82]. This method focuses on the classification of high order digital modulation types [66] and has high resistance to AWGN [17]. It is also more robust against phase and frequency offsets [82]. This method is normally used for FB classification of signals in a multipath fading channel [83], [84], [85]. It is however more computationally complex than features based on the instantaneous amplitude, phase and frequency.

2.2.3 Approach Selection

The following characteristics are proposed as good criteria when evaluating different methods for AMC: versatility, classification accuracy with regards to different noise levels, robustness to channel conditions and computational efficiency [65]. These characteristics are used as guidelines for our evaluation and comparison of AMC algorithms to be used for this study. The main focus of this work is to operate in a non-cooperative environment where many signal- and channel parameters may be unknown. A design based on a classifier that needs perfect channel knowledge becomes logically unsound in a non-cooperative environment where perfect channel knowledge is unattainable. Secondly, the classification algorithm should be suited for hardware implementation and the system is intended to operate as fast as possible with good classification accuracy. A classifier that is costly in terms of time and computation is thus undesirable since computational complexity may impose limitations for hardware implementation. Furthermore, in order to track changes in modulation type of a signal in a non-cooperative environment, a classifier that is able to classify a wide variety of modulation types is needed.

From the literature study above it is evident that likelihood based classifiers are more accurate than feature based classifiers at the expense of computational complexity. The computations are repeated for each modulation hypothesis and each sample. The process is again repeated for a number of iterations when the EM-ML classifier is used. Furthermore, perfect channel knowledge is needed in the case of a ML classifier. Only one or two channel parameters can be unknown in the case of the likelihood ratio test classifiers. The EM-ML classifier is suitable for estimation of multiple unknown channel parameters in a non-cooperative environment; it is however not cost effective in terms of computational complexity.

Because computational cost and operation in non-cooperative environment take precedence for this system, feature based classifiers are rather considered. Features based on the instantaneous amplitude, phase and frequency can be performed relatively quickly without the burden of high computational complexity, making it better suited for hardware implementation. This method is capable of operating in a non-cooperative environment and also has the ability to classify a wide variety of modulation types, including analogue modulations. The features based on higher order statistics are less computationally complex than the LB classifiers. This method can classify a wide range of higher order digital modulation types and is more robust against phase and frequency offsets than features based on the instantaneous information. This method is however significantly more computationally complex than the features based on the instantaneous information.

2.3 Machine Learning and Feature Selection

Machine learning can be used as a decision making process for modulation classification. Machine learning algorithms learn from training data in order to make predictions. These algorithms can become very complex when the number of features they use for decision making is high due to the fact that each feature utilised by the algorithm adds another dimension to the feature space. Feature selection methods are used to select the most useful features for the machine algorithm to optimise classification performance. Computation time can be reduced by the reduction of the feature set and the classification accuracy can be improved. Both feature selection- and machine learning techniques will be discussed below.

2.3.1 Machine Learning

The objective of a machine learning algorithm is to identify an outcome or predict an outcome that is either numeric or categorical. A training dataset is used to train a model in order to fit the data. If a model fits data, it generalises well and does not overfit. The model is then used to predict an outcome based on a set of attributes, known as features, from a new input.

Generalisation is how well a model performs with unseen data, and a test dataset can be used to evaluate its generalisation performance [86]. A model may overfit or underfit a training dataset. Poor generalization performance stems from a machine learning model either overfitting or underfitting the underlying structure in the data [87]. Overfitting occurs when the machine learning algorithm model learns the training data too well and performs poorly for independent test data [87]. With underfitting the opposite occurs. The model is not complex enough and cannot model the training data accurately enough. The complexity of the model is described by its bias-variance decomposition. The bias measures the difference between the average prediction over all datasets and the true mean [88]. The variance measures how much the predictions vary around the true mean for individual datasets and shows how sensitive the model is to a specific dataset [88]. There is always a trade-off between the variance and bias of a model. More complex or flexible models normally have high variance and low bias. These models tend to overfit if the model becomes too complex [89]. More rigid models have low variance and high bias [88]. These models tend to underfit, because they lack the freedom to model the structure of the underlying data [87].

There are three types of learning: Supervised learning, unsupervised learning and reinforcement learning [89]. With supervised learning, the training set consists of input and output sample pairs. Each set of inputs can be mapped to an output label. The system uses these input-output pairs to train a model. The goal is to perform either classification or regression. Classification is the assignment of an input vector to a label or category. The label forms part of a set of finite number of discrete labels [90]. Regression is performed to predict a future value of a continuous variable [90]. For unsupervised learning, the output label is unknown. The system tries to find patterns in the input data and makes use of techniques such as clustering to group input samples or density estimation to ascertain the distribution of data [89], [90]. With reinforcement learning the system learns only from the input data without known output labels, but with reinforcements. When a good decision is made, the system is rewarded and similarly when a bad decision is made, a penalty is given according to a reward function [89]. The system tries to find actions that maximise the reward function [90].

Since classification is the objective, supervised learning algorithms are considered. The most common machine learning techniques for FB classification include Decision trees, Artificial Neural Networks (ANN), Support Vector Machines (SVM) and K-Nearest Neighbours (KNN) [91], [92]. Examples of the utilisation of these machine learning techniques for FB classification in literature are summarised in [91].

Decision trees take a vector of feature values as input and return a single value, known as a decision, as output [89]. The tree makes the decision based on rules inferred from the feature values. The tree consists of nodes and branches. Each node is a test and each branch connected to the node is the outcome of the test. A branch can either have another split or a leaf node. A branch has a leaf node when there is no other test to be performed and a class label can be assigned. The leaf node therefore represents or equates to a class label. The paths from the root to the leaf nodes represent classification rules. The splits are chosen such that each split results in purer branches [93]. A split is pure when it contains branches with only leaf nodes. The best features, selected by a feature selection algorithm, are thus used first in order to find the shortest paths to class labels and therefore results in the shallowest tree possible [89]. The tree can also be pruned to prevent overfitting and improve the classification accuracy [92]. Figure 3 shows an illustration of a decision tree, where the grey box is the root, blue diamonds indicate a node, and the circles represent a leaf node where a classification is made.

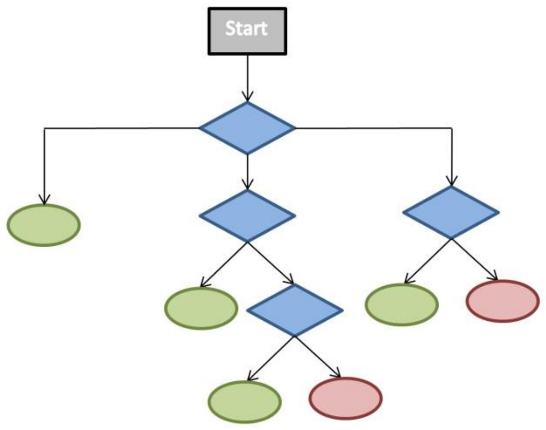


Figure 3: Illustration of a Decision Tree Model

Decision trees are known for their simplicity and is seen as one of the most successful and powerful machine learning techniques because of their classification performance obtained given the simplicity of the trees [89], [94]. Decision trees have the advantage that it is fast to train, and quick to classify data samples when compared to other methods [94]. Decision trees are also accurate for a wide range of classes [94]. Decision trees can be upgraded to classify more class labels by simply adding more branches [91]. Furthermore, no preparation of data is required before it is utilised by the decision tree [94] and it can operate with a combination of numeric and categorical features as well as missing values [95]. They are also robust against outliers [95]. The decision making process of a decision tree is also easier for humans to comprehend [89]. Feature selection forms part of the training process for decision trees. This characteristic makes them resistant to the utilisation of irrelevant features [95]. Branches of decision trees are however based on hard splits where thresholds have to be selected [96]. A decision is made by one node only at a time. The node consists of the selected threshold, which may be a local optimal decision, but may not be a global optimal decision [96]. Decision trees also have high variance and thus tend to overfit. A small change in data can result in a completely different tree structure with different splits [97]. The latter two problems can be addressed by using the decision trees in an ensemble [94], [97].

Artificial neural networks (ANN) are inspired by the behaviour of the human brain. Multiple highly complex, non-linear computations are done in parallel. These complex computations can however be broken down into very simplistic components. These components are known as neurons. The neurons are constructed from the same computational function, however each has its own unique weights and biases and by combining the neurons, the algorithms can become a powerful computational algorithm [93]. ANNs have three types of layers: an input layer, hidden layers and an output layer. A network can have multiple hidden layers where each layer consists of nodes. The nodes are connected to nodes in other layers through weighted links, which propagate the activation from the current node to the next node. The weight of each link determines the sign and strength of each link [89]. The sum of the weighted inputs is computed at each node and an activation function is used to derive an output for the node [89]. The activation function can either be a hard threshold, known as a perceptron, or a logistic function, known as a logistic perceptron [89]. The final value at an output node represents the class label. Back propagation is used for training of neural networks. The classification error is calculated for an output and gets propagated back through the neural network. The weights of the links are then modified in order to minimise the error [92]. Figure 4 shows an illustration of a three layer neural network.

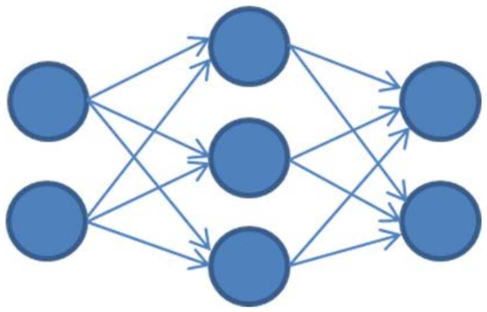


Figure 4: Illustration of a three layer Neural Network Model

ANNs are known for their high classification accuracy and the ability to generalise well [92]. The flexibility of the nonlinear nodes in a neural network enables the network to learn and model relationships of complex data [91]. Since these algorithms aim to find the best values for the link weights, the learning method of these values can also be configured [93]. Furthermore, these models can be very compact which leads to faster computation than algorithms with similar generalisation performance such as SVM [88]. More nodes and layers can be added to increase the accuracy performance of the algorithms as well as to classify a larger number of classes [93]. The increase in the number of nodes and layers however leads to higher computational costs. Another drawback of ANN is that outputs may yield a local optimum which may not be the global optimal solution [91].

The problem of local optimums of both decision trees and ANN techniques are overcome by support vector machines [91]. The function that determines the parameter values in the model is convex which means it has a single global optimum [89], [98]. Given labelled training data, SVMs aim to find the maximum distance between classes by finding a hyperplane in a feature space with the largest margin between the different classes [93]. This is accomplished by finding the hyperplane that maximises the distance between the hyperplane and the nearest data points on both sides of the plane [99]. The hyperplane can then be used to classify new data samples. Support vectors are used to determine the hyperplane. The support vectors are training samples closest to the hyperplane and therefore also the most difficult to distinguish. When the largest margin between these support vectors are found, the hyperplane can maximise the distance between classes [93]. The margin is calculated by the perpendicular distance from the dataset's closest point [98]. For data that cannot be separated with a linear function the inputs are mapped to a higher-dimensional feature space where a linear separator can be found by means of a kernel function [89], [91]. An illustration of a linear SVM model is shown in Figure 5.

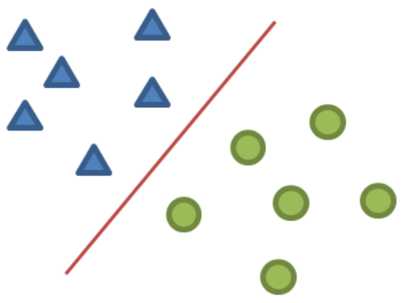


Figure 5: Illustration of a Support Vector Machine Model

Advantages of SVMs include their capability of processing high-dimensional data with few parameters needed [92] and their accuracy due to the convex objective function [88]. After these models are trained, their calculation speed is also much less than other techniques such as ANN. Only one dot product has to be calculated for every new input [89], [99]. Furthermore, SVMs only allow binary classification. If more than two classes are present, the algorithm constructs multiple SVMs in order to classify between one label and the rest of the labels [91].

The last machine learning technique to consider is KNN. This technique does not require a training phase like the previous machine learning techniques. The training data is stored for the prediction phase [92]. The class label of an output is determined by a number of K nearest neighbouring samples to the new input sample. The majority vote amongst the nearest neighbouring samples is assigned to the new input sample [89]. A distance function such as the Euclidian distance, just one choice among many others, is used to determine the distance between an input sample and each neighbour [93]. The number K is chosen as an odd number to prevent ties [89]. Figure 6 shows an illustration of a KNN model.

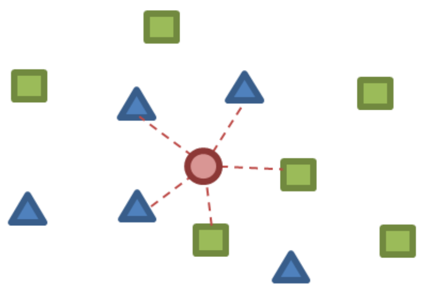


Figure 6: Illustration of a K-Nearest Neighbour Model

This algorithm has high flexibility and is adaptable to multidimensional spaces [87]. It is also robust against outliers [95]. The decision boundaries of KNNs however depend on the input points and their positions. With few input samples, the model may become unstable [87]. Although a training phase is not needed, the computation time for classification is high since all computations are done in the classification stage. The entire training set also needs to be stored and all features are used to compute the distances [89], thus the larger the dataset the more calculations need to be done for each classification.

It can be seen that classification accuracy, computational complexity, ability to generalise and versatility to classify are metrics that were considered in literature when different techniques were evaluated. Considering the advantages and disadvantages discussed in the previous paragraphs, the two comparative studies on machine learning techniques for FB classification, [91] and [92], are used to choose a machine learning technique, for this work.

In [92] a comparative study on Decision Trees, KNN, ANN and SVM has been performed. The authors evaluated the performance of these machine learning techniques for blind FB-AMC under varying SNR by considering the accuracy and complexity of these machine learning techniques. In this study different MIMO configurations and SNR values have been used for the evaluation of the classification accuracy. SVM and ANN showed to have very competitive performance. The accuracy of SVM was however the highest. The overall performance of the four classifiers, under all conditions and configurations, showed to be very close. The computational complexity of decision trees have been shown to be the lowest and SVMs and KNN to be the highest. ANN and SVM are also slower in training than decision trees and KNN. ANN however had the best performance-complexity trade-off [92].

A survey on ANNs, SVMs and decision trees has been performed in [91]. It has also been shown that SVM has the highest classification accuracy and is able to generalise better than the other techniques at lower SNR. It is also mentioned that the design and implementation of decision trees are not complex. If needed, more decision points can be added to the tree to classify more modulation types, without the need of retraining the classifier. This is not possible with the other machine learning techniques. Decision trees are also the most used technique for various FB classification methods in literature investigated in this work [91].

Although the selecting of thresholds limits decision trees, its ability to classify a wide range of modulation types and its simplicity makes it an attractive machine learning technique. SVMs and ANNs are more accurate than decision trees, but their higher computational complexity and slower prediction time than decision trees are drawbacks. KNN's need to store all data points as well as the computation time is also undesirable. As mentioned previously, decision trees are fast to learn as well as making predictions. Its robustness against outliers and ability to select features in the training process are also great advantages over the other machine learning techniques. If necessary, decision trees can be used in ensemble to improve performance by addressing the local optimum and high variance problems. For instantaneous based features, it is evident from [91] and literature (see [100], [101], [102], [103] and [104] for examples) that decision trees are a well-used technique for this class of problem. An improved algorithm with the utilisation of a decision tree shows significant classification accuracy in [100]. The algorithm was tested against varying SNR as well phase shift. It is thus shown that the performance of decision trees can be improved.

2.3.2 Feature Selection

Feature selection aims to reduce a feature set by identifying the most useful features in order to make classifiers more accurate and efficient. Features that provide relevant information are selected while irrelevant and redundant features get eliminated without reducing the accuracy of the classifier. The features are selected based upon a selection criterion which measures the relevance of each feature [105]. Feature selection should not be confused with techniques such as Principle Component Analysis (PCA) where new features are created by combining existing features to reduce the dimensions of the feature space [105]. There are three main methods for feature selection: filter methods, wrapper methods and embedded methods. Filter methods use ranking techniques as criterion. Wrapper methods use search algorithms to find a subset and use the performance of a classifier as criterion. The subset of features that gives the highest classification performance is chosen. For embedded methods, feature selection is incorporated into the training process [105].

As mentioned in the previous section, decision trees select features to split on as part of the training process, which is thus an embedded method. It is therefore not necessary to compare between the three feature selection methods. The feature selection methods used in decision trees as well as the measures for splitting nodes will however be studied.

A decision tree selects the most useful features to split a node. These features are selected based on the node impurity. A feature that will result in the purest branches will be selected first. There are three measures for node impurity: the misclassification error, the Gini Index and the cross-entropy of deviance [97].

For a given node m in a region, R_m, with N_m observations, the portion of observations for class k in node m, is presented by $\hat{p}_{mk}$. The three measures of node impurity, $Q_m(T)$, are then given by:

Misclassification error:

$$Q_m(T) = 1 - \hat{p}_{mk} \tag{9}$$

Gini Index:

$$Q_m(T) = \sum_{k=1}^{K} \hat{p}_{mk}\,(1 - \hat{p}_{mk}) \tag{10}$$

Cross-entropy of deviance:

$$Q_m(T) = -\sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk}) \tag{11}$$

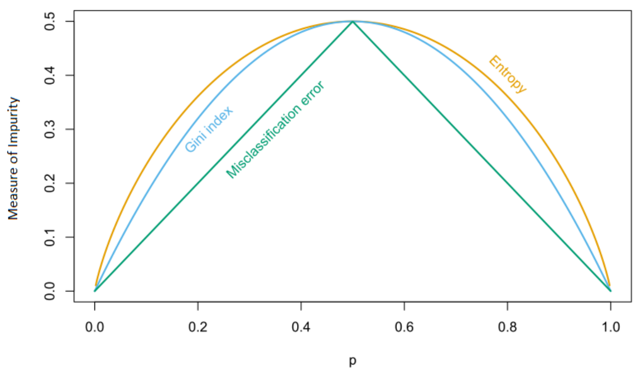


Figure 7: Node Impurity measures [97]

Figure 7 shows the impurity measures as a function of the portion of observations. They are all 0 for $\hat{p}_{mk} = 0$ and $\hat{p}_{mk} = 1$ and have a maximum at $\hat{p}_{mk} = 0.5$. Cross-entropy has been scaled to go through (0.5, 0.5), which does not affect learning. The formation of the tree is thus constructed such that regions contain the highest portion of observations from one class [96].

Gini Index and cross-entropy are differentiable and can be optimised by gradient based optimisation methods [96]. Another advantage is that they have higher sensitivity to node probabilities [97]. These two methods are therefore normally used for growing a tree, where cross-entropy is the most popular method [106]. The misclassification error method is normally used to prune the tree [96], [97].

2.4 Conclusion

In this chapter a literature study has been performed on various techniques in order to find an algorithm for blind modulation classification in a non-cooperative environment. After a suitable signal model was derived, various techniques and methods for classification were discussed and one is chosen based on specific criteria. The criteria were operation in a non-cooperative environment, computation complexity, classification accuracy, and versatility. From the two main approaches for AMC (namely FB and LB classification), FB classification was chosen. This approach is less computationally complex and does not need channel state information. The latter characteristic is vital in a non-cooperative environment where signals with unknown channel parameters may appear. Furthermore, instantaneous based features for FB classification have been chosen for the feature extraction stage, as opposed to wavelet transform and higher order statistics based features. Instantaneous based features are the least computationally complex, suitable for hardware implementation and are able to classify a wide range of modulation types, including the modulation types of interest for this study. For the decision making stage, various machine learning techniques were considered. Decision trees were selected for their simplicity, fast run time and their ability to classify a wide range of modulation types, among other advantages. It is a proven machine learning technique in literature for FB classification tasks and the classification accuracy of this technique is close to other machine learning techniques such as SVMs and ANNS. The computation of the features as well as the construction of the decision tree is performed in the chapters that follow.

3 CONCEPTUAL DESIGN

The results from Chapter 2 are used for the design decisions of the system. Functional flow diagrams are used to explain the process to be followed for classification and tracking of changes in a signal.

In order to track changes in transmitter modulation type, three main steps are required. The first step is to receive RF signals from the environment. This is typically through an antenna. The second step is to perform pre-processing on the received signals. After the necessary steps are taken to obtain the signal of interest and get it in its correct form, the classification of the modulation type and tracking of changes can take place. Figure 8 shows the functional flow block diagram of the main process.

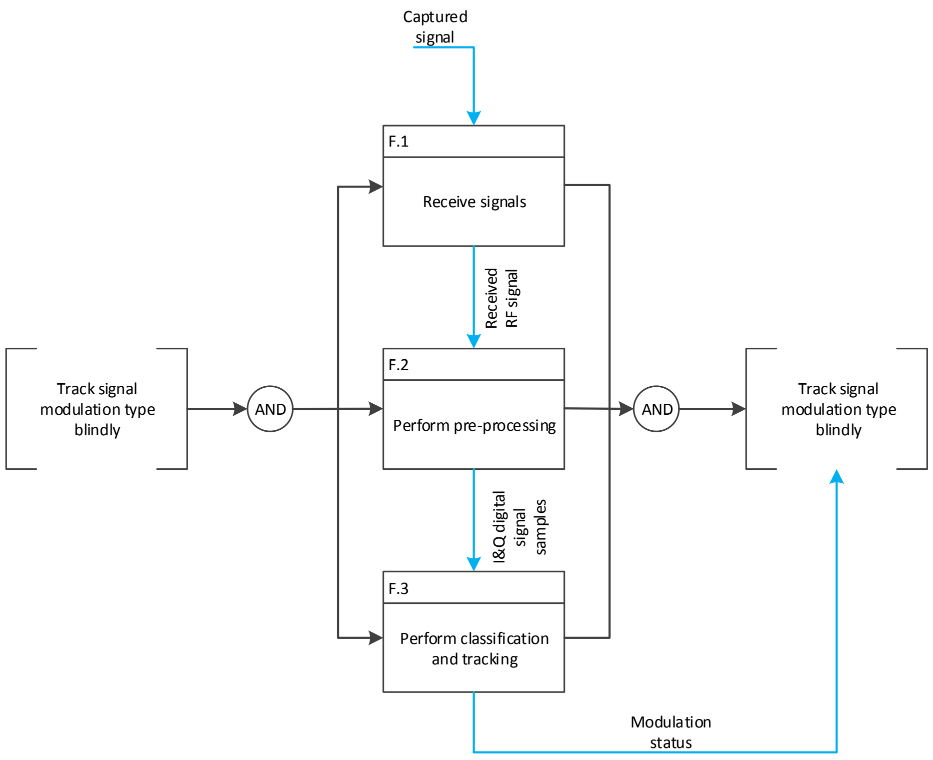

Captured signal
F.1
Receive signals
Received RF signal
Track signal modulation type blindly
AND
F.2
Perform pre-processing
AND
Track signal modulation type blindly
I&Q digital signal samples
F.3
Perform classification and tracking
Modulation status

Figure 8: Top Level Functional Flow Diagram with Focus on Automatic Modulation Classification & Tracking

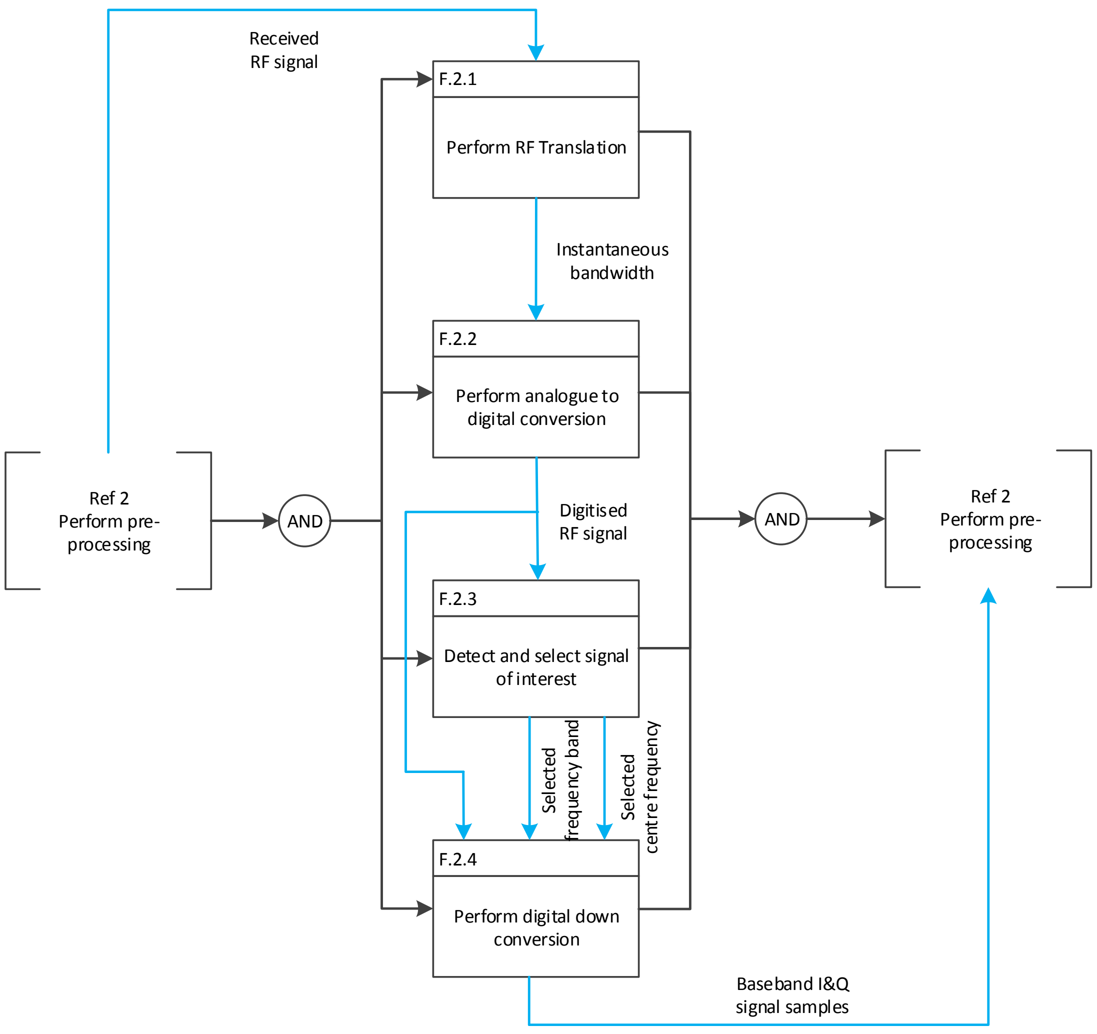
The main process consists of the following functions:

- F.1: Signals in the selected frequency band are received by an antenna.
- F.2: The instantaneous bandwidth (IBW) is captured from the RF signal and processed to I&Q baseband samples by a front-end processor.
- F.3: The I&Q baseband samples of the signal of interest are used to perform the classification and tracking process in order to output the modulation type and any changes from one modulation type to another.

The focus of this study is captured in the third functional block F.3: Perform classification and tracking. An understanding of the design of functional block F.2 is however needed, specifically with regards to the interfaces in order to design the third step correctly. These two steps will be discussed in detail below.

3.1 Front-end Processing

After situational awareness is obtained from the RF band, which contain a wide range of RF signals, the IBW is captured. The IBW is designed to match the band of the analogue to digital converter (ADC). This frequency band contains various intentional, unintentional and unavoidable signal sources, including signals of interest, spurious signals and noise. Pre-processing is performed by the front-end processor of a digital receiver to obtain the signal of interest. General front-end processors include RF translation, analogue-to-digital conversion (ADC), detection and selection of frequency of interest and digital down conversion [107], [108]. The detection of the centre frequency may occur using a phase-locked loop (PLL) or a direct digital frequency synthesizer (DDS). Figure 9 shows the functional flow block diagram of the front-end processing.



Received RF signal
F.2.1
Perform RF Translation
Instantaneous bandwidth
F.2.2
Perform analogue to digital conversion
Ref 2 Perform pre-processing
AND
Digitised RF signal
AND
Ref 2 Perform pre-processing
F.2.3
Detect and select signal of interest
Selected frequency band
Selected centre frequency
F.2.4
Perform digital down conversion
Baseband I&Q signal samples

Figure 9: Functional Flow Diagram of the Pre-processing Functional Block

The pre-processing consists of the following functions:

- F.2.1: RF translation of the received RF signal mixes the signal down into a band where the ADC can sample the signal.
- F.2.2: Analogue to digital conversion is performed to convert the RF down-mixed analogue signal to digital samples.
- F2.3: The band in which the signal of interest resides is detected and the RF band centre frequency is selected using a PLL or DDS. The output of a PLL is locked to a crystal oscillator reference, to provide a stable output frequency that is used for down mixing [107].
- F.2.4: Digital down conversion is performed to obtain the filtered baseband signal in its complex form.

3.1.1 Analogue to Digital Conversion

The IBW is converted to digital samples after the RF translation stage. The sampling frequency is chosen to be at least twice the bandwidth of the RF band of interest to satisfy the Nyquist condition.

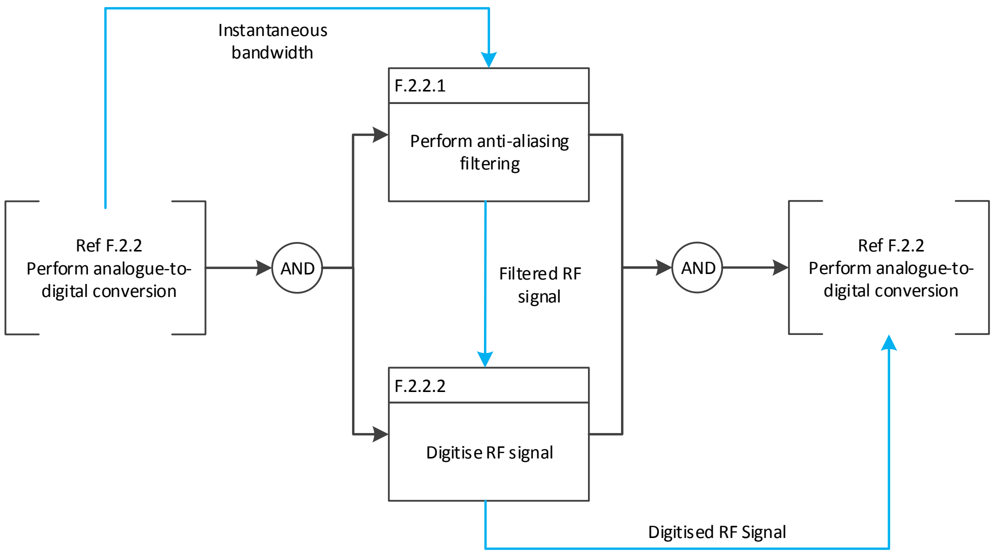


Figure 10: Functional Flow Block Diagram of the Analogue to Digital Conversion Functional Block

The analogue-to-digital conversion block consists of:

- F.2.2.1: A low pass filter is used to remove all signals above $f_s/2$ to prevent aliasing to occur, where f_s is the sampling frequency.
- F.2.2.2: An ADC is used to convert the filtered RF band of interest to digital samples.

3.1.2 Digital Down Conversion

The digital down conversion consists of three main parts: in-phase and quadrature (I&Q) demodulation, low-pass filtering, and decimation. Phase information that needs to be maintained is contained within the I&Q samples of the complex signal. The complex signal is presented by:

$$\tilde{z}(t) = x(t) + jy(t) \tag{12}$$

Where $x(t)$ is the real part and $y(t)$ is the imaginary part of the complex signal, obtained from the real signal through a Hilbert transform. The baseband signal can be obtained from the complex passband signal. The complex baseband signal is given by:

$$\tilde{r}_b(t) = \tilde{r}_p e^{-j2\pi f_c t} \tag{13}$$

such that:

$$\tilde{r}_b(t) = r_{bI}(t) + jr_{bQ}(t) \tag{14}$$

where f_c is the carrier frequency, $\tilde{r}_p(t)$ is the complex received passband signal and $r_{bI}(t)$ and $r_{bQ}(t)$ are the baseband in-phase and quadrature signals respectively [109].

A Hilbert filter or a mixer is used to obtain the complex baseband in-phase and quadrature (I&Q) samples. A mixer uses a sine and a cosine signal which have 90 degrees offset in phase between them. These two signals are generated by a numerically controlled oscillator (NCO) to mix the output from the ADC to either baseband or an intermediate frequency (IF). The outputs of the mixer are complex digital samples consisting of in-phase and quadrature (I&Q) components [107], [108]. The functional flow block diagram of the digital down conversion is shown in Figure 11.

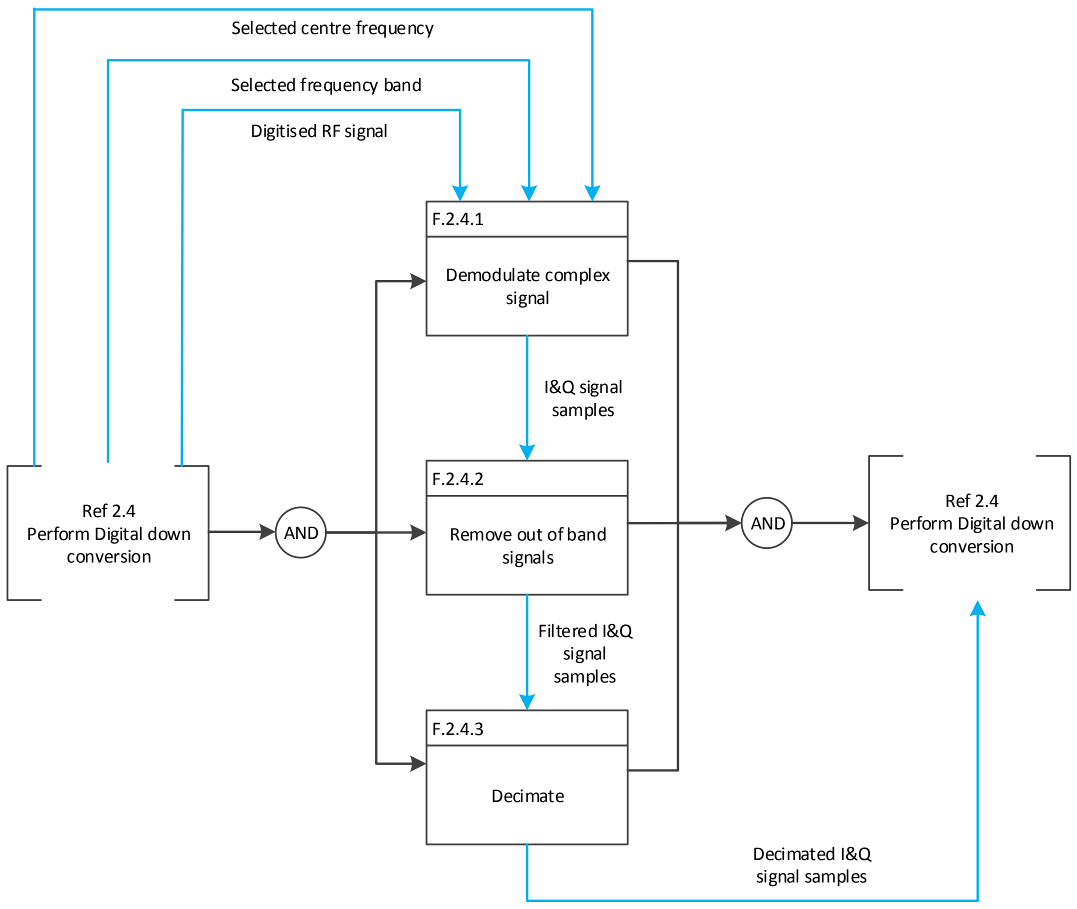

Selected centre frequency
Selected frequency band
Digitised RF signal
F.2.4.1
Demodulate complex signal
I&Q signal samples
F.2.4.2
Remove out of band signals
Filtered I&Q signal samples
F.2.4.3
Decimate
Decimated I&Q signal samples
Ref 2.4
Perform Digital down conversion
AND
AND
Ref 2.4
Perform Digital down conversion

Figure 11: Functional Flow Block Diagram of Digital Down Conversion Functional Block

The digital down conversion is performed by:

- F.2.4.1: A Hilbert filter or an NCO and mixer are used to obtain the complex representation of the real input. The outputs of the mixer are baseband I&Q signals samples.
- F.2.4.2: A low pass filter is used to pass the baseband I&Q samples. Frequencies above the selected cut-off frequency are filtered out.
- F.2.4.3: Decimation is performed to reduce the sampling rate and bandwidth. A reduced sampling rate at the lower bandwidth increases the effective processing power available to process the resulting signal.

3.2 Classification and Tracking

The classification and tracking functional block receives the I&Q baseband samples in order to determine the modulation type of the signal as well as track changes from one modulation type to another. Two main steps are required, a classification and a tracking process. The functional flow block diagram is shown in Figure 12. After these two steps are taken, the system can output the status of the signal's modulation type.

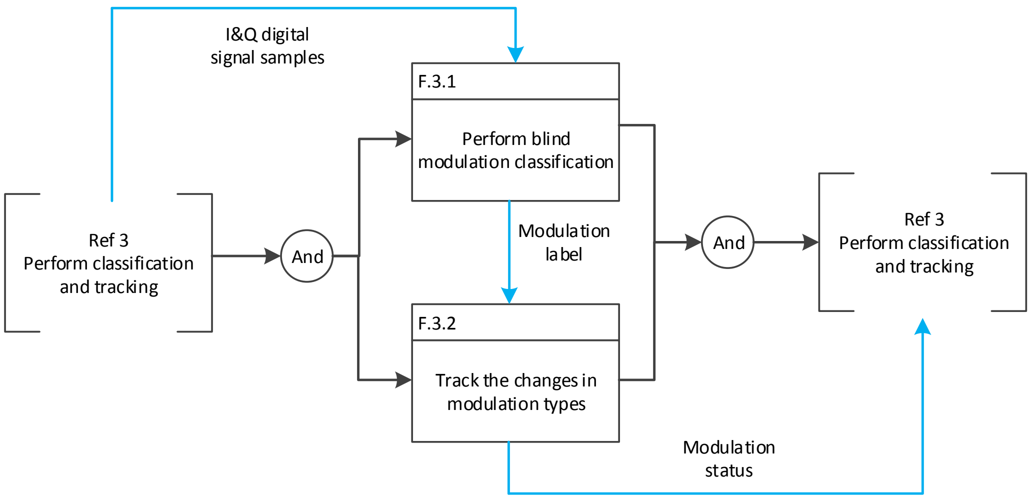

I&Q digital signal samples
F.3.1
Perform blind modulation classification
Ref 3
Perform classification and tracking
And
Modulation label
And
Ref 3
Perform classification and tracking
F.3.2
Track the changes in modulation types
Modulation status

Figure 12: Functional Flow Block Diagram of the Classification and Tracking Process

The classification and tracking functional block consists of:

- F.3.1: Features are extracted from the I&Q samples to perform blind classification where the output is a modulation class label.
- F.3.2: The tracking of changes of the modulation type is performed by means of logging the modulation type over time, in order for external processes to utilise the status of the modulation type.

Within this design blind modulation classification is performed through two distinct functional steps: feature extraction and decision making. In Chapter 2, instantaneous features were selected for feature extraction, and a decision tree classifier was selected as the machine learning technique for decision making. The functional flow block diagram illustrating these functions is shown in Figure 13.

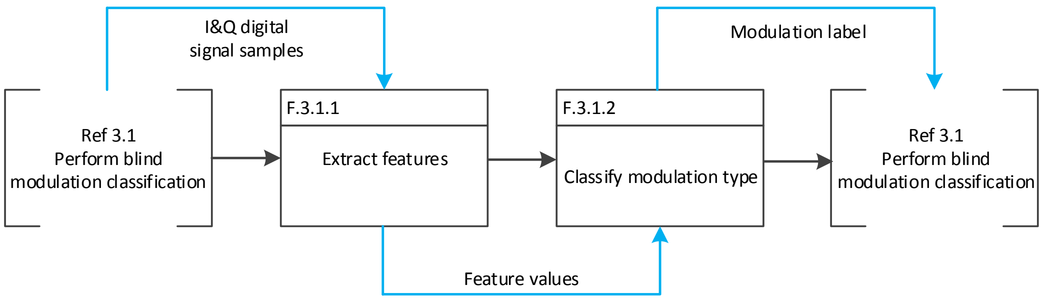

I&Q digital signal samples
Modulation label
F.3.1.1
F.3.1.2
Ref 3.1 Perform blind modulation classification
Extract features
Classify modulation type
Ref 3.1 Perform blind modulation classification
Feature values

Figure 13: Functional Flow Block Diagram of the Blind Modulation Classification Functional Block

The blind modulation classification is performed by:

- F.3.1.1: Features are extracted from the I&Q input samples by using the instantaneous information of the signal.
- F.3.1.2: The feature values are used to classify the modulation type of the intercepted signal using a decision tree classifier. The output of the decision tree is a modulation class label.

3.2.1 Feature Extraction

For feature extraction, the instantaneous amplitude, phase and frequency are calculated from the complex signal given in (12). Thereafter, features based on the instantaneous amplitude, phase and frequency are extracted. The functional flow block diagram illustrating these functions is shown in Figure 14.

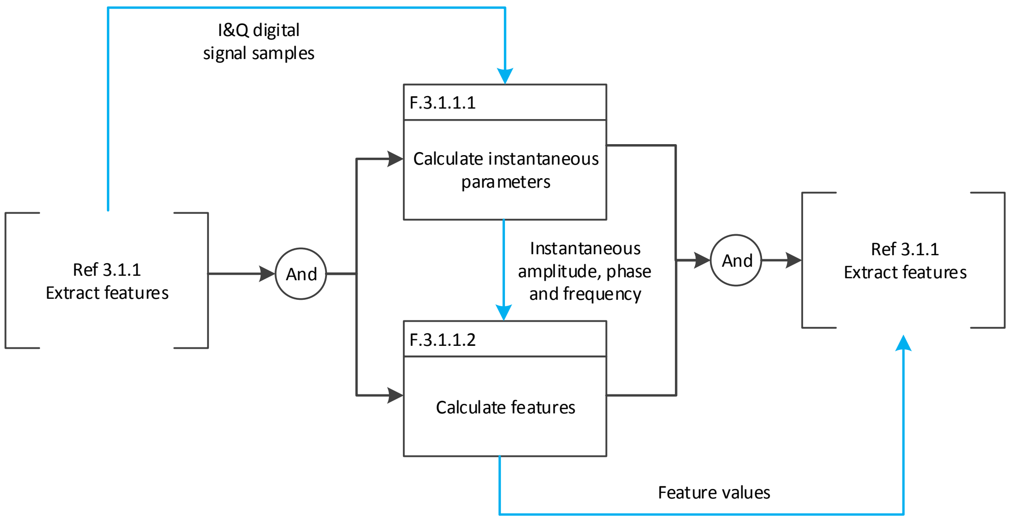

I&Q digital signal samples
F.3.1.1.1
Calculate instantaneous parameters
Ref 3.1.1 Extract features
And
Instantaneous amplitude, phase and frequency
And
Ref 3.1.1 Extract features
F.3.1.1.2
Calculate features
Feature values

Figure 14: Functional Flow Block Diagram of the Feature Extraction Functional Block

The feature extraction is performed by:

- F.3.1.1.1: The instantaneous amplitude, phase and frequency is calculated from the complex signal consisting of the I&Q baseband samples.
- F.3.1.1.2: Features are calculated by using the instantaneous information of the received baseband signal.

The instantaneous amplitude, phase and frequency are calculated from the complex signal (12). The polar form of the complex baseband signal is given by:

$$\tilde{r}_b(t) = A(t)e^{j\phi(t)} \tag{15}$$

where $A(t)$ and $\phi(t)$ are the instantaneous amplitude and phase respectively. The instantaneous amplitude is the magnitude of the complex I&Q signal, while the instantaneous phase is the angle of the signal. The instantaneous frequency is the derivative of the instantaneous phase. The instantaneous amplitude, phase and frequency are given by (16), (17) and (18) respectively over N number of samples at time instants $t = \frac{i}{f_s}$ with $i = 1,2, \dots, N$ where f_s is the sample frequency [109].

$$A[i] = |\tilde{r}[i]| = \sqrt{{r_{bI}}^2[i] + {r_{bQ}}^2[i]} \tag{16}$$

$$\phi[i] = \angle\tilde{r}[i] = \tan^{-1}\left[\frac{r_{bQ}[i]}{r_{bI}[i]}\right] \tag{17}$$

$$f[i] = \frac{1}{2\pi}\frac{d\phi(t)}{dt} = \frac{1}{2\pi}\left[\frac{\phi[i] - \phi[i-1]}{T_s}\right] \tag{18}$$

where

$$T_s = \frac{1}{f_s} \tag{19}$$

Before the feature values are calculated, the instantaneous amplitude and frequency are centred and normalised to compensate for channel gain. Additionally, the centred non-linear component of the instantaneous phase is obtained.

$$A_{cn}[i] = \frac{A[i]}{\mu_A} - 1 \tag{20}$$

$$f_{cn}[i] = \frac{f[i] - \mu_f}{f_s} \tag{21}$$

$$\phi_{NL}[i] = \phi[i] - \mu_\phi \tag{22}$$

where μ_A, μ_ϕ and μ_f are the averages of the instantaneous amplitude, phase and frequency respectively.

Using the equations above, the extraction of the instantaneous amplitude, phase and frequency can be determined. The calculation of the instantaneous amplitude and phase in hardware is less time consuming when using the polar form instead of the rectangular form of a signal. The calculation from the rectangular form includes computations such as division and square root, shown in (16) and (17), which may take multiple clock cycles and cause bottlenecks in the processing chain. The use of these complex functions should be kept to a minimum for hardware implementation due to computational resource limitations. The rectangular form of the signals is thus first converted to the polar form, before the instantaneous amplitude, phase and frequency are obtained. Figure 15 shows the functional flow block diagram to obtain the instantaneous amplitude, phase and frequency.

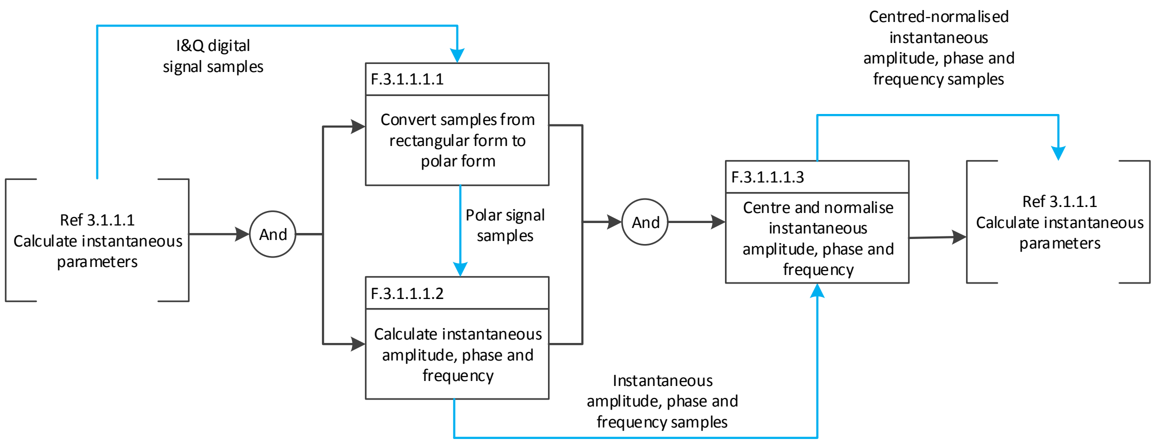

I&Q digital signal samples
F.3.1.1.1.1
Convert samples from rectangular form to polar form
Polar signal samples
F.3.1.1.1.2
Calculate instantaneous amplitude, phase and frequency
Ref 3.1.1.1 Calculate instantaneous parameters
And
And
Instantaneous amplitude, phase and frequency samples
F.3.1.1.1.3
Centre and normalise instantaneous amplitude, phase and frequency
Centred-normalised instantaneous amplitude, phase and frequency samples
Ref 3.1.1.1 Calculate instantaneous parameters

Figure 15: Functional Flow Block Diagram to Obtain Instantaneous Amplitude, Phase and Frequency

The instantaneous information is obtained by:

- F.3.1.1.1.1: The rectangular form of the signal, represented by the I&Q samples, are converted to polar form.
- F.3.1.1.1.2: The instantaneous amplitude and phase are obtained from the absolute value and angle of the polar signals samples respectively. The instantaneous phase is used to calculate the instantaneous frequency.
- F.3.1.1.1.3: The instantaneous amplitude, phase and frequency are centred and normalised.

After the instantaneous information is calculated, the features can be extracted. There are eight general instantaneous based features to consider [110], [111]. The eight features are given in (23) to (25) and (27) to (31). Implementation on hardware is taken into consideration when features are selected. Features with lower computational cost, even at the cost of some accuracy, are better suited to hardware platforms such as FPGAs. Seven features from the eight features were selected based on this consideration. The eighth feature (31) was not selected because it requires an FFT to be calculated, which makes the computational cost of this feature high. It was therefore replaced with another amplitude based feature shown in equation (26) [112]. Four features are derived from the instantaneous amplitude, two features from the instantaneous phase and the last two features from the instantaneous frequency.

The four amplitude based features are the standard deviation of the absolute value of the centred-normalised instantaneous amplitude (23), the standard deviation of the centred-normalised instantaneous amplitude over the non-weak intervals of the signal (24), the kurtosis of the centred-normalised instantaneous amplitude (25) and the mean of the centred-normalised instantaneous amplitude (26). The non-weak intervals are not sensitive to noise and can be detected by a threshold A_t. The detection by means of a threshold is explained in more detail in Chapter 4.

$$\sigma_{aa} = \sqrt{\frac{1}{N}\left(\sum_{i=1}^{N} A_{cn}^{2}\,[i]\right) - \left(\frac{1}{N}\sum_{i=1}^{N} |A_{cn}[i]|\right)^{2}} \tag{23}$$

$$\sigma_a = \sqrt{\frac{1}{N_c}\left(\sum_{A_n[i]>A_t} a_{cn}^2[i]\right) - \left(\frac{1}{N_c}\sum_{A_n[i]>A_t} a_{cn}[i]\right)^2} \tag{24}$$

where N_c is number of samples for which $A_n[i] > A_t$.

$$\mu_{42}^a = \frac{E\{A_{cn}^4[i]\}}{\left\{E\{A_{cn}^2[i]\}\right\}^2} \tag{25}$$

$$A_{mean} = \frac{1}{N}\sum_{i=1}^{N}|A_{cn}[i]| \tag{26}$$

The two features based on the instantaneous phase are the standard deviation of the absolute value of the centred non-linear component of the instantaneous phase (27), and the standard deviation of the centred non-linear component of the direct instantaneous phase (28) calculated over the non-weak intervals of the signal.

$$\sigma_{ap} = \sqrt{\frac{1}{N_c}\left(\sum_{A_n[i]>A_t} \phi_{NL}^2[i]\right) - \left(\frac{1}{N_c}\sum_{A_n[i]>A_t} |\phi_{NL}[i]|\right)^2} \tag{27}$$

$$\sigma_{dp} = \sqrt{\frac{1}{N_c}\left(\sum_{A_n[i]>A_t} \phi_{NL}^2[i]\right) - \left(\frac{1}{N_c}\sum_{A_n[i]>A_t} \phi_{NL}[i]\right)^2} \tag{28}$$

The two frequency based features are the standard deviation of the absolute value of the centred-normalised instantaneous frequency (29) and the kurtosis of the centred-normalised instantaneous frequency (30).

$$\sigma_{af} = \sqrt{\frac{1}{N_c}\left(\sum_{A_n[i]>A_t} f_{cn}^2[i]\right) - \left(\frac{1}{N_c}\sum_{A_n[i]>A_t} |f_{cn}[i]|\right)^2} \tag{29}$$

$$\mu_{42}^f = \frac{E\{f_{cn}^4[i]\}}{\left\{E\{f_{cn}^2[i]\}\right\}^2} \tag{30}$$

The maximum value of the spectral power density of the centred-normalised instantaneous amplitude is given by:

$$\gamma_{max} = max|DFT(A_{cn}[i])|^2/N_c \tag{31}$$

3.3 Conclusion

The decisions of Chapter 2 were used to perform the conceptual design for tracking changes in emitter modulation type. This chapter provided insight of the pre-processing required to obtain the signal of interest for feature extraction. The method for obtaining feature values as well as the method to utilise the feature values for decision making were discussed. It is shown that the instantaneous amplitude phase and frequency of the signal is obtained and used to calculate the feature values. For the decision making step, a decision tree is first trained and then used to classify and track changes in modulation types. The conceptual design can be used in Chapter 4 in which the implementation of the feature extraction and decision making process are explained in detail.

4 IMPLEMENTATION AND RESULTS

The conceptual design in the previous chapter was used to develop a system capable of classifying six digital modulation types and tracking changes between modulation types. The design was first simulated in Matlab after which an FPGA firmware design was implemented. Several experiments were performed in both simulation and hardware to evaluate the performance of the system. In the next section the generation of the input signals used for testing is described. Thereafter, both simulation and hardware implementation of the system are described in detail. The experiments performed are explained and the results are presented and discussed.

4.1 Signal Generation

For the simulation of the six digitally modulated signals, a message signal was first generated. Channel effects were then added to the modulated signals as required by the various experiments performed. More specifically, a sequence of N_S random, independent integers, $m = 0,1,\dots,M-1$, was generated to create the levels of the message signal given by:

$$\theta(t) = \sum_{i}^{N_S} m_i p(t - nT) \tag{32}$$

Where T is the bit duration of the i_{th} integer and $p(t)$ is a rectangular pulse given by:

$$p(t) = \begin{cases} +1 & for\ 0 \le t \le T \\ 0 & otherwise \end{cases} \tag{33}$$

Equation (1) was used to derive equation (34) for the six digitally modulated signals:

$$s_\theta(t) = A_\theta \cos(2\pi f_\theta t + \phi_\theta) \tag{34}$$

Communication systems have definite bandwidths in which they operate. In order to create more realistic test signals, a band pass filter was used to band limit the generated signals. Bandwidths were selected in accordance to the proposed modulation type. The band limitation was thus performed after the generation of the modulated signal. The bandwidth (B_ω) contains 97.5% of the total average power of the signal [113].

$$\int_{f_c - B_\omega/2}^{f_c + B_\omega/2} G_s(f)df = 0.975 \int_{-\infty}^{\infty} G_s(f)df \tag{35}$$

The centre frequency (f_c), symbol rate (r_s) and sample frequency (f_s) were chosen as 150 kHz, 12.5 kBd and 1200 kHz respectively for all signal modulation types. These parameters were set to the same values selected in [110] in order to aid in comparison. The amplitude, phase and frequency for each modulation type as well as the bandwidths of the modulated signals are shown in Table 1.

Table 1: Signal Parameters

Modulation Type	A_θ	ϕ_θ	f_θ	**Bandwidth**
2ASK	$0.8\theta + 0.2$	0	f_c	$4r_s$
4ASK	$0.25\theta + 0.25$	0	f_c	$4r_s$
2PSK	1	$\pi\theta + \frac{\pi}{2}$	f_c	$6r_s$
4PSK	1	$\frac{\pi}{2}\theta$	f_c	$6r_s$
2FSK	1	0	$f_c + 4r_s\theta - 2r_s$	$8r_s$
4FSK	1	0	$f_c + r_s\theta$ $(m = -3, -1, 1, 3)$	$8r_s$

For the additive noise, a sequence of real numbers with Gaussian distribution and zero mean was generated. The size of the sequence of numbers was equal to N_s . A bandpass filter with a bandwidth related to the intended modulated signal was used to filter the noise. In practice, the bandwidth of a receiver is normally slightly larger than the bandwidth of the intercepted signal. The bandwidth was therefore chosen as 1.2 times the bandwidth of the signal. The desired SNR in decibels was obtained by multiplying the band-limited noise sequence $\{n(i)\}$ with a coefficient R_{sn}, which is the ratio of the signal power S_p to noise power N_p [67]:

$$R_{sn} = \sqrt{\frac{S_p}{N_p}\left[10^{\frac{-SNR}{20}}\right]} \tag{36}$$

$$S_p = \sum_{i=1}^{N} s^2(i) \tag{37}$$

$$N_p = \sum_{i=1}^{N} n^2(i) \tag{38}$$

with $i = 1,2, \dots, N_s$.

For the fading channel, multipath delays and path gains were chosen such that they result in the desired delay spread as discussed in Chapter 2. Equation (40) was used to determine the delay spread values. The power of the kth multipath signal is given by:

$$P(\tau_k) = a_k^2 \tag{39}$$

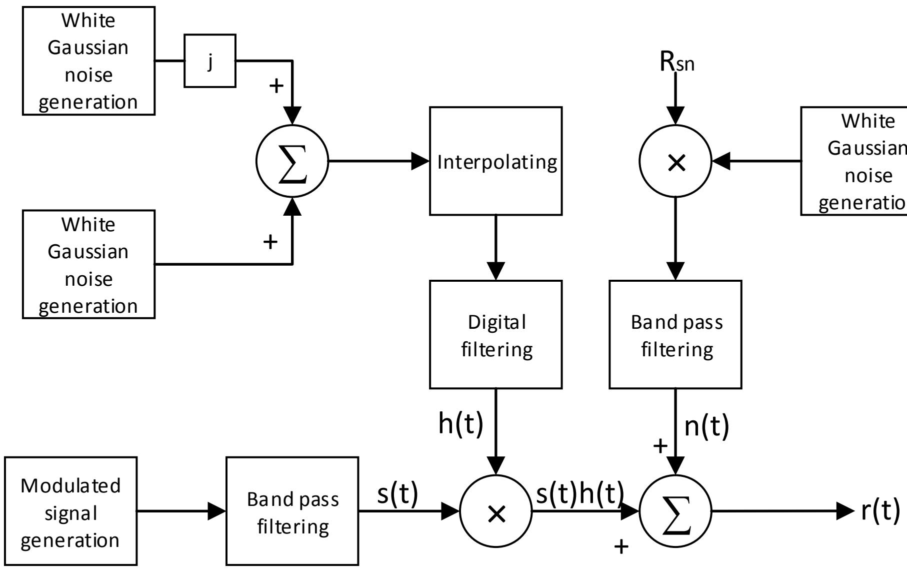
where τ_k is the excess delay and a_k the amplitude.

By substituting (39) into (6) the path delays and path gains could be selected to give the desired delay spread:

$$\sigma_T = \sqrt{\frac{\sum a_k^2 \tau_k^2}{\sum a_k^2} - \left(\frac{\sum a_k^2 \tau_k}{\sum a_k^2}\right)^2} \tag{40}$$

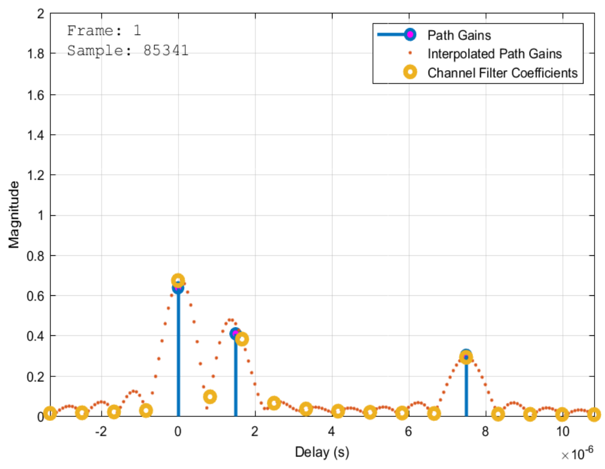
The process of generating the input signals is shown in Figure 16. The impulse response and frequency response of a static Rayleigh fading channel simulated in Matlab is shown in Figure 17 and

Figure 18 respectively. These figures display examples of a single instance; the impulse and frequency response of the channel vary from test signal to test signal.



White Gaussian noise generation
j
+
White Gaussian noise generation
+
Interpolating
Rsn
White Gaussian noise generation
Digital filtering
Band pass filtering
h(t)
n(t)
+
Modulated signal generation
Band pass filtering
s(t)
s(t)h(t)
+
r(t)

Figure 16: Flow Diagram of Signal Generation [114], [24]



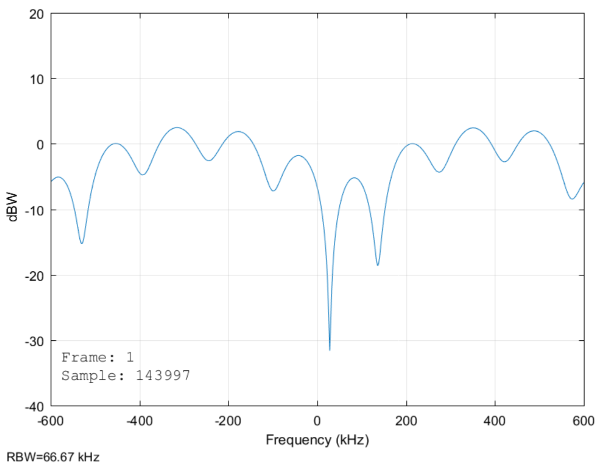


Figure 17: Impulse Response of a three-path Rayleigh Fading Channel Simulated in Matlab

Figure 18: Frequency Response of a three-path Rayleigh Fading Channel Simulated in Matlab

4.2 Matlab Simulation

In this section the simulation of the feature extraction, classification and tracking are described. The calculation of the instantaneous amplitude, phase and frequency is first explained. It is followed by the calculation of the feature values and construction of the decision tree used for classification of the modulation types. The last part discusses the tracking of changes in the modulation type. The experiments that were performed for the different parts of the system are then presented and described.

4.2.1 Implementation

4.2.1.1 Instantaneous amplitude, phase and frequency

The classification and tracking algorithm used the signals generated in section 4.1 as inputs. The complex baseband in-phase and quadrature signals were first obtained. To correspond with the filter that was pre-implemented on the hardware platform, a Type 3 Hilbert FIR filter of order 30 was first used to obtain the Hilbert transform of the received signal. The filter, with frequency response shown in Figure 19, has unity gain and linear phase. The Hilbert filter is described in more detail in section 4.3.1.1.1.

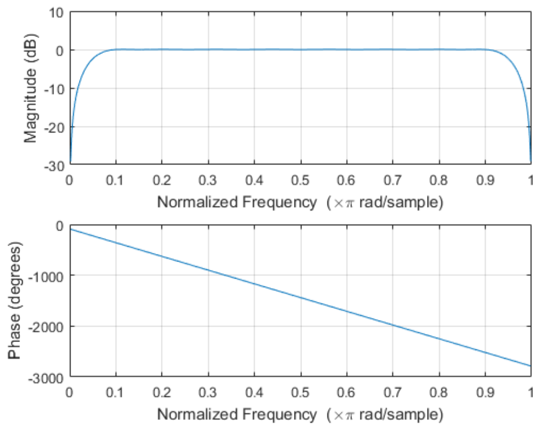


Figure 19: Frequency Response of a Type 3 Hilbert FIR Filter

The baseband in-phase and quadrature signals were then obtained by using equations (13) and (14). The front-end processor for hardware implementation is capable of determining the carrier frequency and estimation of the carrier frequency is thus not required. For the Matlab simulation, equations (41) and (42) were used to mix the input signal to baseband in-phase and quadrature signals [109].

$$r_{bI}(t) \ = x(t)\cos(2\pi f_c t) + y(t)\sin(2\pi f_c t) \tag{41}$$

$$r_{bQ}(t) \ = y(t)\cos(2\pi f_c t) - x(t)\sin(2\pi f_c t) \tag{42}$$

where $x(t)$ and $y(t)$ are defined in (12). For the calculation of the instantaneous amplitude, phase and frequency, equations (16) to (18) were used. With the calculation of the instantaneous phase, the phase is constrained by its principal value and is called the wrapped phase. The principal value is in the range $(-\pi, \pi]$ and has discontinuities of 2π radians when viewed as a function of the radian frequency [115]. It can be corrected by adding multiples of $\pm 2\pi$ to ensure a continuous function of phase. The unwrapping can be mathematically described as:

$$C_p(i) = \begin{cases} C_p(i-1) - 2\pi & if\ \phi(i+1) - \phi(i) > \pi \\ C_p(i-1) + 2\pi & if\ \phi(i) - \phi(i+1) > \pi \\ C_p(i-1) & elsewhere \end{cases} \tag{43}$$

where $\{C_p(i)\}$ is the phase correction sequence and $C_p(i) = 0$. The unwrapped phase can then be given by:

$$\phi_{uw} = \phi(i) + C_p(i) \tag{44}$$

After the angle of the complex signal was calculated, the *unwrap* function of Matlab was used to obtain the unwrapped instantaneous phase [116]. The difference between two consecutive samples of the unwrapped phase was then used to calculate the instantaneous frequency. The process for the calculation of the instantaneous amplitude, phase and frequency in Matlab simulation is shown in Figure 20.

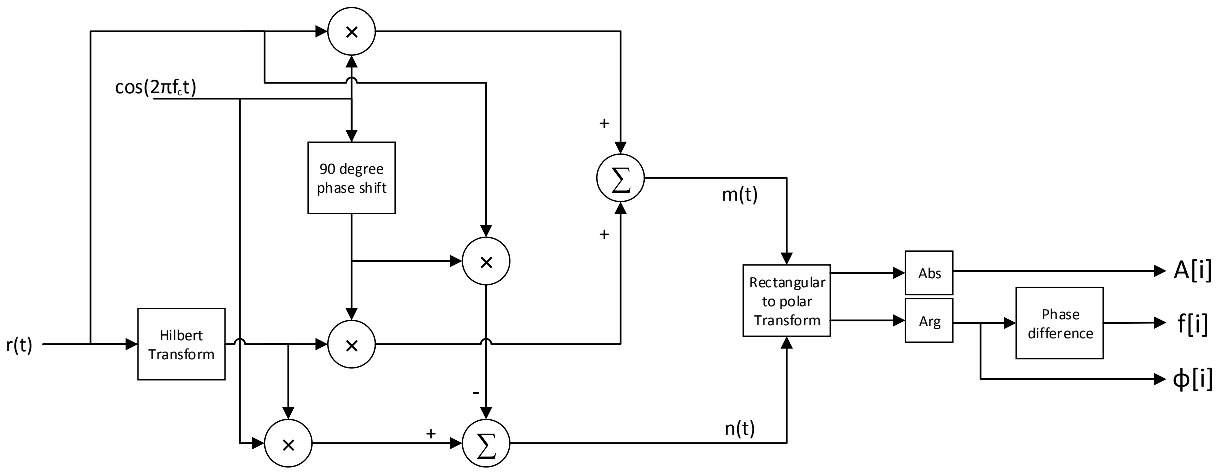


Figure 20: Flow Diagram for Calculation of Instantaneous Amplitude, Phase and Frequency in Matlab Simulation

Figure 21 to Figure 26 show the instantaneous amplitude, phase and frequency of the six different modulation types.

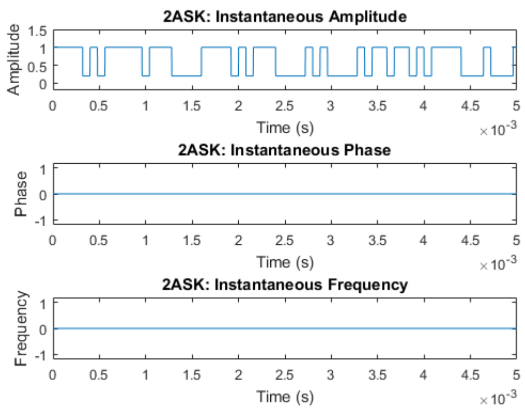


Figure 21: Behaviour of 2ASK over Time

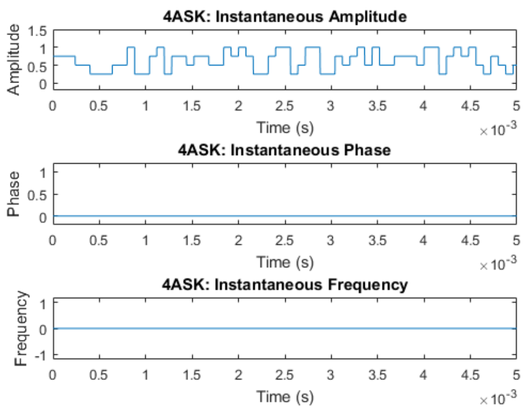


Figure 22: Behaviour of 4ASK over Time

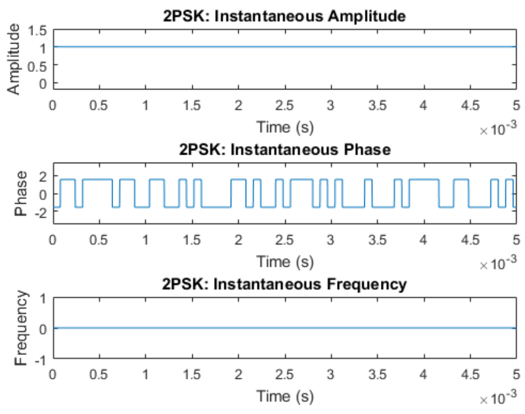


Figure 23: Behaviour of 2PSK over Time

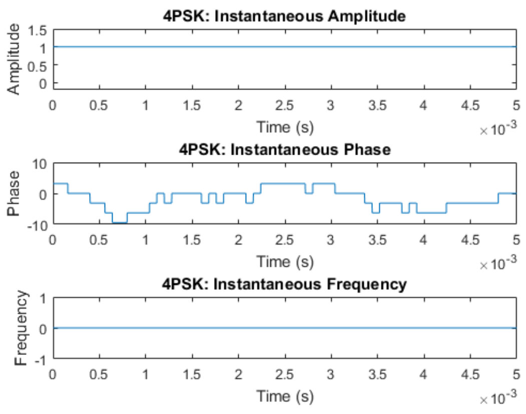


Figure 24: Behaviour of 4PSK over Time

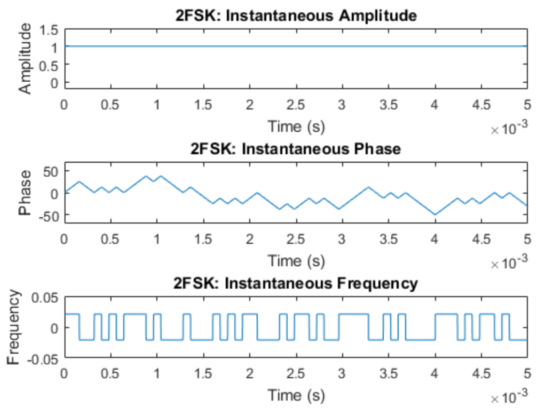


Figure 25: Behaviour of 2FSK over Time

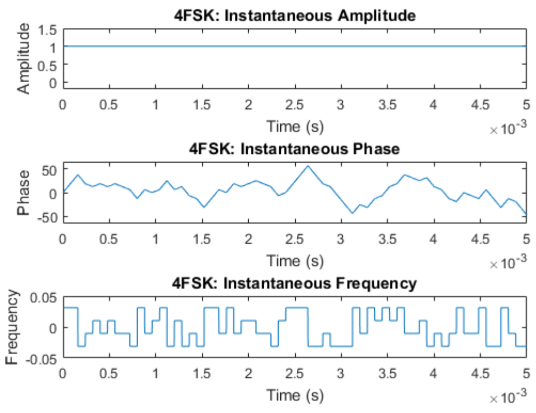


Figure 26: Behaviour of 4FSK over Time

From the figures it can be seen that the bit stream is contained in the instantaneous amplitude for MASK. For MPSK the bit stream is contained in the instantaneous phase and for MFSK the bit stream is contained in the instantaneous phase and frequency.

After the instantaneous information was obtained, equations (20) to (22) were used to calculate the centred-normalised amplitude and frequency, and the centred non-linear phase. A received passband signal contains an undesired linear phase component mainly contributed by the carrier frequency. The non-linear phase can be obtained by:

$$\phi_{NL}(i) = \phi_{uw} - \frac{2\pi f_c i}{f_s} \tag{45}$$

Since the signal is at baseband when these calculations are performed, the linear component is not present and the phase only needs to be centred at zero using equation (22).

The next step was to determine the weak intervals of the signal. The weak intervals are found where there are phase transitions for MPSK (which can also be seen in Figure 33 and Figure 34). These parts of the signal are sensitive to noise [110]. The weak intervals of the signal can be detected by evaluating the amplitude against a threshold.

The authors of [110] evaluated the normalised amplitude values against a threshold and only made use of the non-weak intervals of the signal to calculate σ_a, σ_{ap}, σ_{dp} and σ_{af}. This method was not used here for two reasons: The number of samples might become too few if they were removed. The second reason is that the calculations of the features which are dependent on the evaluation of the amplitude samples can only start after the normalised amplitude is calculated. In order to find the normalised amplitude, the average of the amplitude has to be calculated. This will be time consuming and undesirable for hardware calculations.

The following method was used instead to compensate for the weak amplitude values. After the centred-normalised amplitude was calculated, the amplitude samples were evaluated against a threshold and a constant value was assigned to the amplitude values that exceeded the threshold. From Figure 21 to Figure 26 it can be seen that A_{cn} of MPSK and MFSK should be zero and for MASK $|A_{cn}| < 0.8$. The amplitude threshold was thus set to 0.8. Amplitude values that exceeded the threshold were set to 0.

A threshold method was also used for the instantaneous frequency to compensate for the transition effects of MPSK. The transition effects will be explained in section 4.2.2.1. In Figure 21 to Figure 26 it can be seen that all frequency values should be $|f_{cn}| < 0.05$. The threshold was thus set to 0.05. Frequency values that exceeded the threshold were set to 0. The process to calculate the centred-normalised amplitude, centred non-linear phase and centred-normalised frequency is shown in Figure 27.

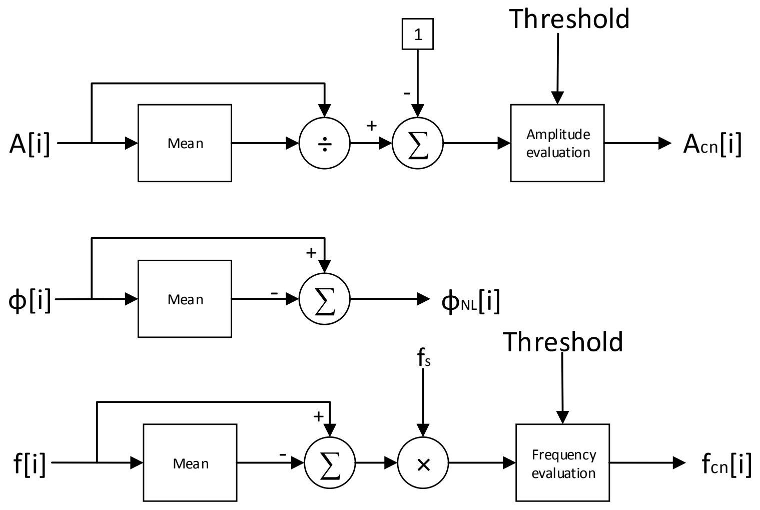


Figure 27: Flow Diagram for Normalising and Centring the Instantaneous Amplitude, Phase and Frequency in Matlab Simulation

4.2.1.2 Instantaneous based features

The centred-normalised amplitude, centred non-linear phase and centred-normalised frequency were used as inputs to the algorithms presented in Figure 28, Figure 29 and Figure 30 respectively. The feature values were calculated using equations (23) to (30).

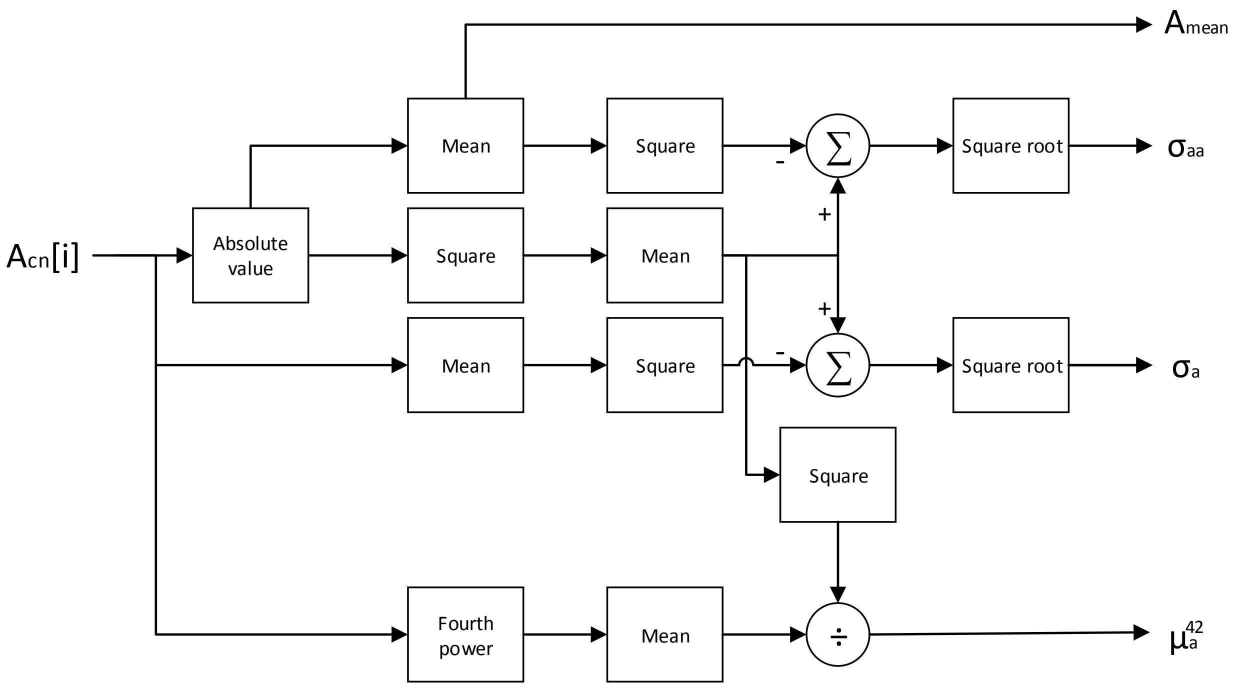


Figure 28: Flow Diagram for Instantaneous Amplitude Based Feature Extraction in Matlab Simulation

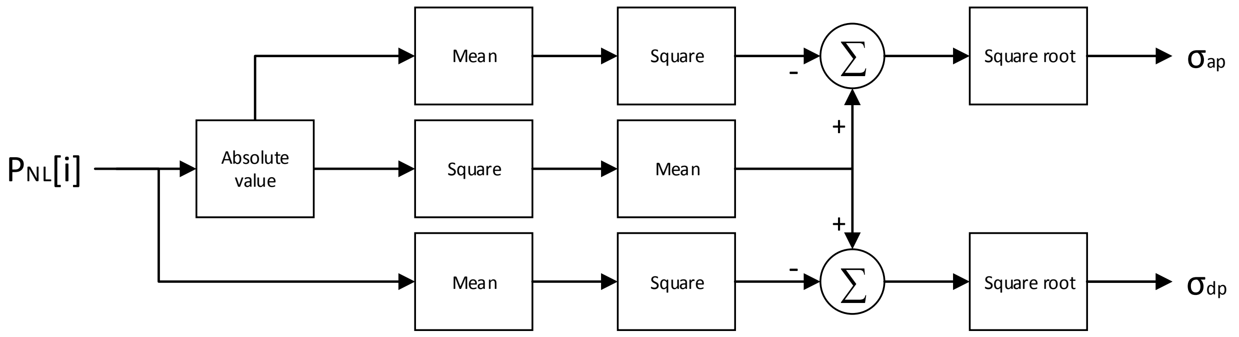


Figure 29: Flow Diagram for Instantaneous Phase Based Feature Extraction in Matlab Simulation

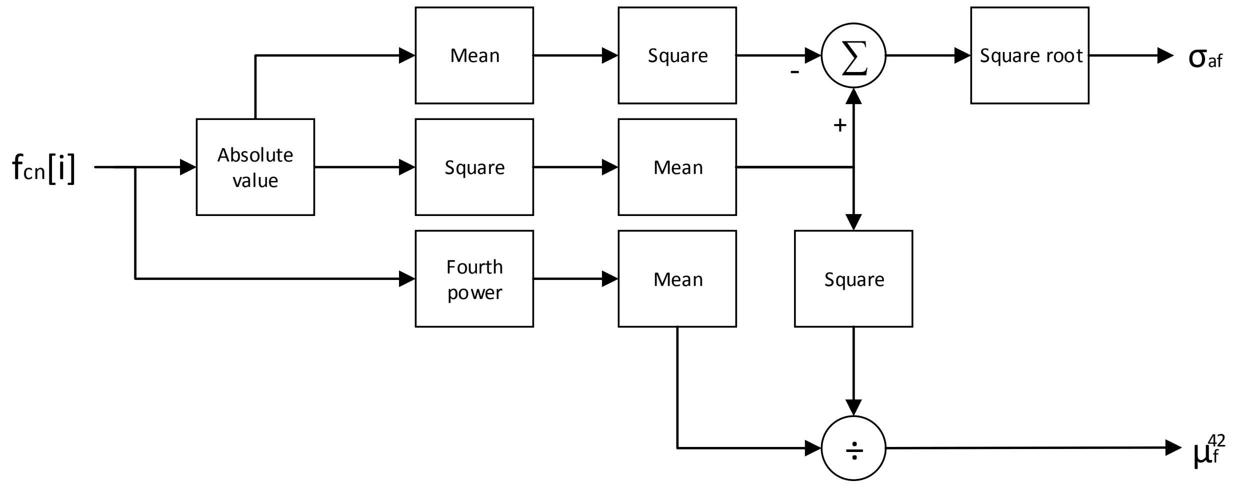


Figure 30: Flow Diagram for Instantaneous Frequency Based Feature Extraction in Matlab Simulation

4.2.1.3 Decision Tree Construction

The decision tree was constructed using the Matlab *fitctree* object from the Statistics and Machine Learning Toolbox [117]. The function returns a binary classification decision tree which is based on the input features in a matrix X and output labels in matrix Y. The branches of the tree were split on the nodes based on the values of the input matrix X. One of three tests for selecting the best features to split nodes as well as the criteria for splitting the nodes could be specified.

The three tests for feature selection are: the standard CART (Classification and Regression Tree) test, the curvature test and the interaction-curvature test. The CART test selects the feature which maximises the split criterion gain over all the possible splits among all the features [118]. The curvature test selects the feature which minimises the p-value of the chi-square test of independence between each feature and the class label [119]. The curvature-interaction test performs the curvature test, and additionally performs the test between each pair of features and the class label to prevent the selection of irrelevant features. The CART test is not sensitive to interactions between features and important features are less likely to be selected [120]. The curvature-interaction test, which test for both interaction between a features, and features and class labels, was thus selected. For the split criterion, the cross-entropy for node splitting criteria was also selected as discussed in Chapter 2.

The tree was first grown and cross validation was then used to determine the best level to prune the tree to. The eight feature values obtained from the feature extraction calculations were concatenated into a [1x8] vector to form part of the input matrix X. The matrix X is the input training dataset consisting of multiple [1x8] vectors. The corresponding modulation types were used to form matrix Y consisting of the training labels. The unpruned tree's classification error of the training set was determined by the re-substitution error using the *resubLoss* function [121]. The classification error is calculated by:

$$Classification\ error = \frac{\#\ misclassifications}{Total\ \#\ observations} \tag{46}$$

The classification error of the validation set was determined by cross-validation error using the *cvloss* function [121]. The tree overfits the training data if the re-substitution error is significantly smaller than the cross-validation error and the tree needs to be pruned. The best pruned level of the tree is then determined by the *cvloss* function. The function determines the tree for which the cross-validation error is a minimum and returns the level that is within one standard error of the minimum. The *prune* function is then used to prune the tree to the best level.

The *predict* function was used to determine the output of the decision tree. This function uses the constructed classification tree and vector X containing the feature values of the incoming signal as input and returns the predicted modulation type.

4.2.2 Results

Various experiments were performed to investigate the performance of the algorithm for classification and tracking of transmitter modulation types. The different steps of the method were investigated separately and thereafter as a whole. The experiments include the calculation of the instantaneous amplitude, phase and frequency, the extraction of features from signals under varying SNR and fading conditions, the classification accuracy of the decision tree and the feature extraction from signals with decreasing signal lengths under varying SNR and fading conditions.

4.2.2.1 Instantaneous Amplitude, Phase and Frequency

The instantaneous amplitude, phase and frequency for each modulation type were first investigated in order to determine the degree to which the instantaneous amplitude, phase and frequency influence the measured features. The investigation was done by comparing the calculated instantaneous amplitude, phase and frequency to the expected theoretical instantaneous amplitude, phase and frequency.

Figure 31 to Figure 36 show the instantaneous amplitude phase and frequency of the six different modulation types calculated from the received passband signal. No noise and no fading had been added to these signals.

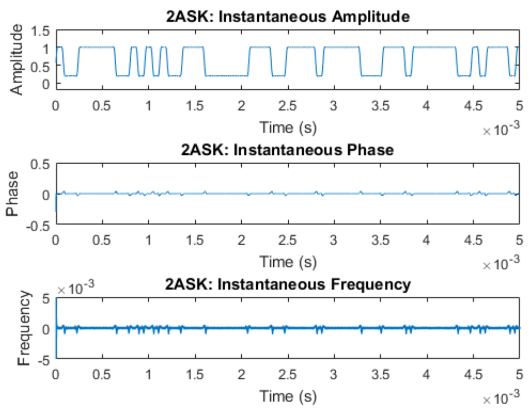


Figure 31: Calculated Instantaneous Amplitude, Phase and Frequency of 2ASK

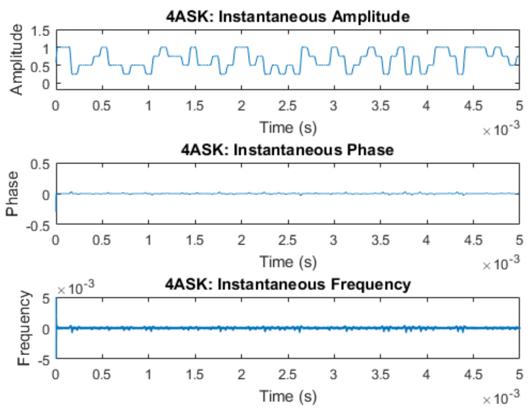


Figure 32: Calculated Instantaneous Amplitude, Phase and Frequency of 4ASK

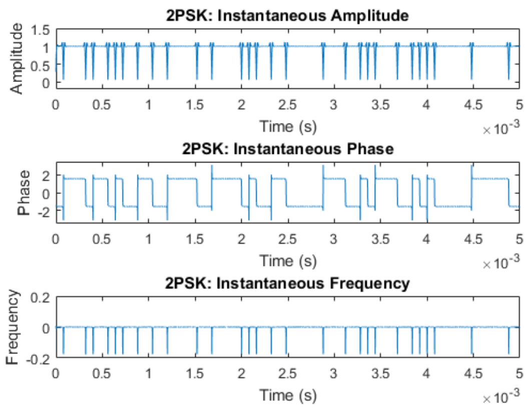


Figure 33: Calculated Instantaneous Amplitude, Phase and Frequency of 2PSK

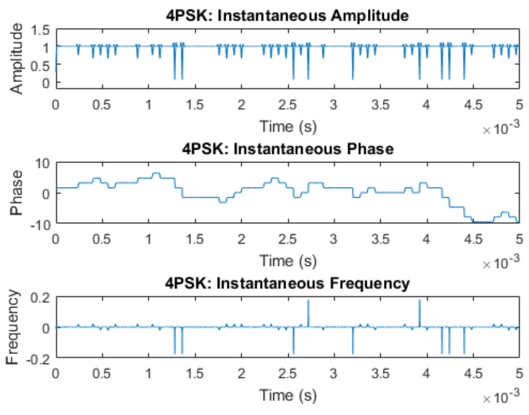


Figure 34: Calculated Instantaneous Amplitude, Phase and Frequency of 4PSK

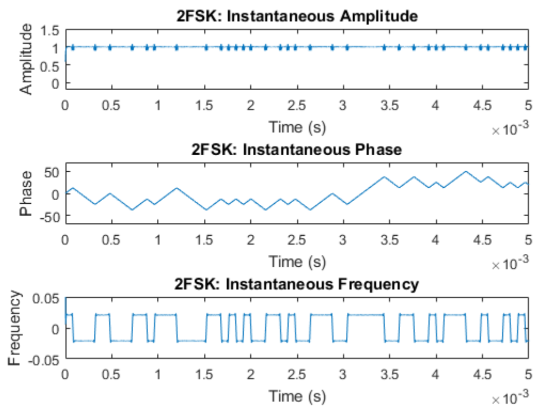


Figure 35: Calculated Instantaneous Amplitude, Phase and Frequency of 2FSK

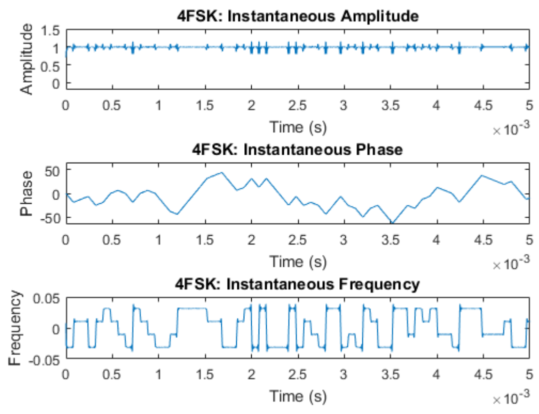


Figure 36: Calculated Instantaneous Amplitude, Phase and Frequency of 4FSK

From these figures, it can be seen that there were differences between the theoretical instantaneous information shown in Figure 21 to Figure 26 and calculated instantaneous information. The effect of the band limitation was evident for all six modulation types. The band limitations also caused variations in the amplitude of MPSK and MFSK at the boundaries of symbol transitions, where for the theoretical case the amplitude of MPSK and MFSK is completely flat. The calculation of the derivative of the phase to obtain the frequency caused fluctuations in frequency at symbol transitions of MPSK. The fluctuations in the frequency of MASK were also evident because of the small variations in the phase of MASK, where for the theoretical case the phase of MASK is completely flat. As discussed in section 4.2.1.1, these effects can influence the performance of the system since these fluctuations can be falsely perceived as information in a modulation type's instantaneous amplitude, phase or frequency. Some of these effects were compensated for by the threshold techniques explained in section 4.2.1.1 and the results are shown in Figure 37 to Figure 44.

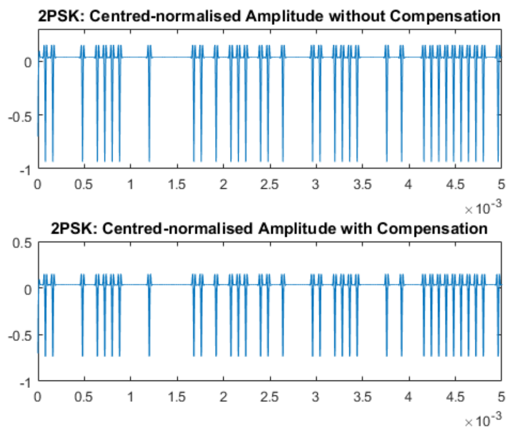


Figure 37: Centred-normalised Amplitude of 2PSK before and after Adjustments

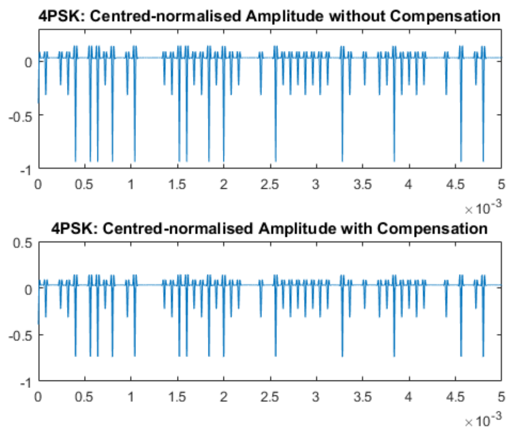


Figure 38: Centred-normalised Amplitude of 4PSK before and after Adjustments

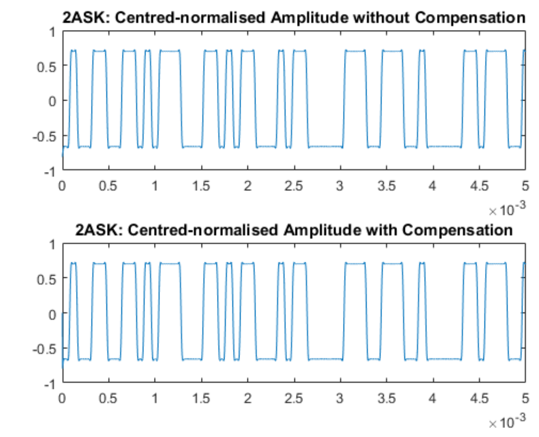


Figure 39: Centred-normalised Amplitude of 2ASK before and after Adjustments

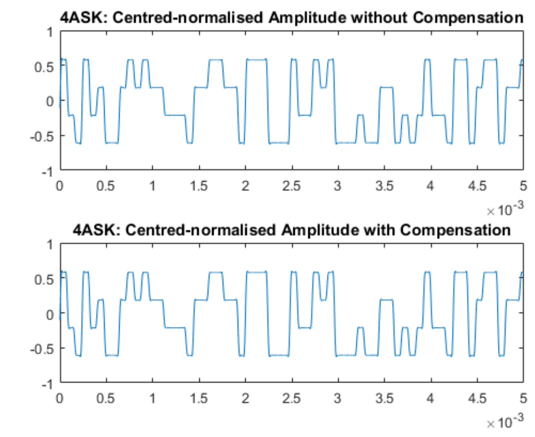


Figure 40: Centred-normalised Amplitude of 2PSK before and after Adjustments

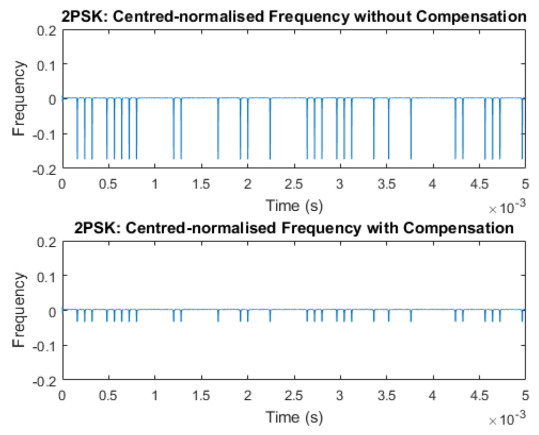


Figure 41: Centred-normalised Frequency of 2PSK before and after Adjustments

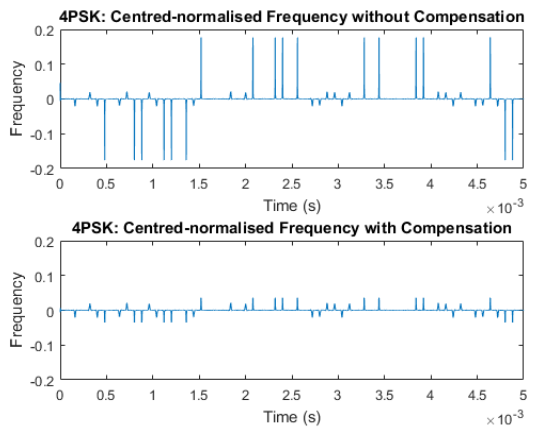


Figure 42: Centred-normalised Frequency of 4PSK before and after Adjustments

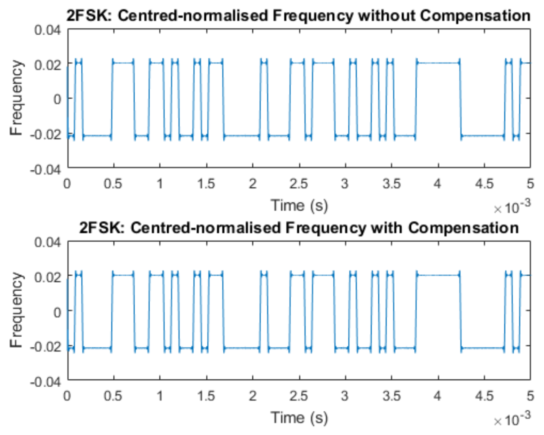


Figure 43: Centred-normalised Frequency of 2FSK before and after Adjustments

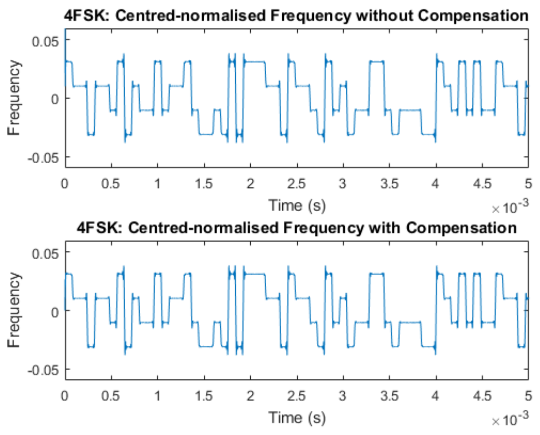


Figure 44: Centred-normalised Frequency of 4FSK before and after Adjustments

It can be seen that the fluctuations in the amplitude of MPSK was compensated for to some extent, while the values of MASK were not affected. The variations in MFSK were too small to be compensated for. The fluctuations in the frequency of MPSK were also significantly compensated for, while the values of MFSK were not affected. The variations in MASK were too small to be compensated for.

4.2.2.2 Comparison without AWGN and fading

Since the carrier frequency, symbol rate, sampling frequency, and features were adopted from [110], an experiment was performed to compare the calculated feature values to the values obtained by the authors of [110]. The experiment was performed for signal lengths of 0.12 seconds (144 000 samples), and 2048 samples, the latter matching the experimental setup of the authors. The experiment was performed for a longer signal (144 000 samples) in order to observe whether there was a significant difference in values when more samples were used. An unrealistically long signal of length 1 second was first simulated, and shortened in steps of 0.2 seconds each time comparing the feature values of the signals in the presence of AWGN and flat fading channel conditions. This comparison was done in order to find a sufficient signal length that exhibited the same characteristics that could be stored for reuse in tests. The stored signals could then be decreased through Matlab operations to a desirable signal length for further analysis as required.

For baseline data, signals without any channel effects were used to calculate the feature values. Five hundred iterations were run for each modulation type. In each iteration, a new message signal was generated and modulated with the six different modulating signals. The averages of the five hundred feature values for each modulation type were calculated. Table 2 shows the values obtained by the authors of [110] and Table 3 and Table 4 show the results of values calculated in Matlab. The features were explained in detail in section 3.2.1.

Table 2: Feature Values obtained using 2048 Samples in [110]

Modulation Type	A_{mean}	σ_{aa}	σ_a	μ_{42}^a	σ_{ap}	σ_{dp}	σ_{af}	μ_{42}^f
2ASK	NA	0.00	0.5	1.5	0.03	0.03	0.00	1.0
4ASK	NA	0.32	0.4	1.8	0.03	0.03	0.00	1.0
2PSK	NA	0.00	0.1	2.2	0.304	1.57	0.10	3.6
4PSK	NA	0.00	0.1	2.8	4.77	6.67	0.13	3.7
2FSK	NA	0.00	0.0	1.0	6.39	9.47	0.06	1.4
4FSK	NA	0.00	0.0	1.0	5.62	8.50	0.48	1.7

Table 3: Calculated Feature Values of Noise-free Signals over 0.12 second

Modulation Type	A_{mean}	σ_{aa}	σ_a	μ_{42}^a	σ_{ap}	σ_{dp}	σ_{af}	μ_{42}^f
2ASK	0.6216	0.1357	0.6363	1.0848	0.0231	0.0237	0.0002	303.1075
4ASK	0.3758	0.2012	0.4263	1.7307	0.0220	0.0223	0.0001	512.5669
2PSK	0.0739	0.1124	0.1338	16.7709	0.1358	1.5749	0.0032	34.6618
4PSK	0.0680	0.0923	0.1144	17.3388	1.0785	1.9018	0.0048	18.5439
2FSK	0.0147	0.0230	0.0273	12.0843	0.8992	1.8122	0.0033	1.0471
4FSK	0.0198	0.0295	0.0355	11.8130	0.9016	1.8186	0.0105	1.6803

Table 4: Calculated Feature Values of Noise-free Signals over 2048 Samples

Modulation Type	A_{mean}	σ_{aa}	σ_a	μ_{42}^{a}	σ_{ap}	σ_{dp}	σ_{af}	μ_{42}^{f}
2ASK	0.5727	0.1776	0.5991	1.2809	0.1669	0.1669	0.0006	2.5412
4ASK	0.3730	0.2151	0.4307	1.8708	0.1672	0.1682	0.0006	2.7227
2PSK	0.0847	0.130	0.1297	15.9728	0.3780	1.5170	0.0030	29.6284
4PSK	0.0817	0.0864	0.1162	14.3290	0.9209	1.5885	0.0043	15.5510
2FSK	0.0293	0.0214	0.0311	50.8905	0.9095	1.8065	0.0057	1.3264
4FSK	0.0351	0.0285	0.0384	15.8721	0.9155	1.8067	0.0107	1.8108

It can be seen that the values were not identical to the values obtained in [110]. There might be several reasons for these differences. The first reason might be due to different transition effects experienced by the authors of [110] and also the different technique they used to remove the weak intervals of the signal as discussed in 4.2.1.1. The authors of [110] also used different methods to obtain the complex signal, the instantaneous phase and the instantaneous frequency as shown below.

The phase is calculated by:

$$\phi(t) = \begin{cases} \tan^{-1}\left[\frac{y(t)}{x(t)}\right] & if\ x(t) > 0, y(t) > 0 \\ \pi - \tan^{-1}\left[\frac{y(t)}{x(t)}\right] & if\ x(t) < 0, y(t) > 0 \\ \frac{\pi}{2} & if\ x(t) = 0, y(t) > 0 \\ \pi + \tan^{-1}\left[\frac{y(t)}{x(t)}\right] & if\ x(t) < 0, y(t) < 0 \\ \frac{3\pi}{2} & if\ x(t) = 0, y(t) < 0 \\ 2\pi - \tan^{-1}\left[\frac{y(t)}{x(t)}\right] & if\ x(t) > 0, y(t) < 0 \end{cases} \quad (47)$$

where $x(t)$ and $y(t)$ are defined in (12).

Both the complex signal and frequency were calculated by means of the FFT. For the complex representation of signal, the spectrum $X(f)$ of the real signal $x(t)$ was obtained. The spectrum of the complex signal was then calculated by:

$$Z(f) = 2U(f)X(f) \quad (48)$$

where $U(f)$ is the unit step function in the frequency domain and is given by:

$$U(f) = \begin{cases} 1\ if\ f > 0 \\ \frac{1}{2}\ if\ f = 0 \\ 0\ otherwise \end{cases} \quad (49)$$

The complex signal is then obtain by:

$$z(t) = IFFT\{Z(f)\} \tag{50}$$

For the calculation of the frequency, the FFT was used to obtain the derivative of the phase in order to avoid numerical differentiation in the time domain and to obtain smoother results:

$$f(t) = IFFT\{-\mathrm{j}2\pi f\Phi(f)\} \tag{51}$$

where $\Phi(f)$ is the Fourier transform of $\phi(t)$.

In order to reduce the computational complexity for hardware implementation, these three methods were not followed and the techniques described in section 4.2.1.1 were used instead. It is also worth noting that the method used in this study for the calculation of the instantaneous frequency is dependent on the sample frequency. There are several methods for the normalisation of the instantaneous frequency, including the utilisation of the symbol rate and sample frequency. The authors of [110] used the symbol rate for normalisation, which was not chosen for normalisation in this study as the symbol rate of the intercepted signal might be unknown and must then be estimated first. The sample frequency was therefore used instead for normalisation of the instantaneous frequency.

The feature values however showed resemblance when compared. As discussed below, the modulation types that have information in their instantaneous amplitude, phase and frequency had corresponding values for the associated features. The results from the output of the features were promising on initial inspection with regard to the separability of the modulation types. The feature values obtained under different SNR conditions, which will be shown in section 4.2.2.3, also correspond to the results presented in [69] and [123] which provide further confidence that other authors in this field of study followed similar approaches, making our results later in this study directly comparable to their work.

From the feature values it can be observed that:

For A_{mean}

- MASK had larger values than the other modulation types. When the absolute values of the amplitude are centred at zero, the modulation types with no amplitude information have values close to zero. Since the bit stream is contained in the amplitude of the signal for MASK, it contains amplitude information.

For σ_{aa}

- 4ASK had the largest value among all the modulation types. The other modulation types have constant amplitude values except for 2ASK which has two levels. When these values are centred at zero, the absolute values are almost equal and the variance of the absolute amplitude results in values close to zero. Since 4ASK has four levels, the absolute values are not equal and the variance shows that it contains information in the absolute amplitude.

For σ_{ap}

- 4PSK had the largest value among all the modulation types. 2ASK and 4ASK have constant phase values, and 2PSK has two levels. When these values are centred at zero, the absolute values are almost equal and the variance of the absolute phase results in values close to zero. The variance shows that MFSK and 4PSK contain information in their absolute phase.

For σ_{af}

- 4FSK also had the largest value for among all the modulation types. The other modulation types have constant frequency values (zero) and 2FSK has two levels. When these values are centred at zero, the absolute values are almost equal and the variance of the absolute frequency results in values close to zero. Since 4FSK has four levels, the absolute values are not equal and the variance shows that it contains information in the absolute frequency.

For σ_a

- MASK had the larger values than MPSK and MFSK. 2ASK and 4ASK have information in the instantaneous amplitude and thus have larger values than the modulation types with no information in their instantaneous amplitude.

For σ_{dp}

- MPSK and MFSK had larger values than MASK. MPSK and MFSK have information in the instantaneous phase and thus have larger values than MASK with no information in the instantaneous phase.

For μ_{42}^a and μ_{42}^f

- The compactness of the distribution of the instantaneous amplitude and frequency are measured with μ_{42}^a and μ_{42}^f respectively. A large value is related to a wide distribution and a small value is related to a narrow distribution. The instantaneous frequency for MASK and MPSK should theoretically be zero, since they contain no frequency information. The band limitations and transition effects however caused different results. From Figure 31 to Figure 34 it can be seen that the fluctuations caused the distribution of the values to seem wide and the kurtosis thus had a much larger value than the theoretical case.

By comparing the results of the features calculated over 0.12 seconds (144 000 samples) and 2048 samples, it can be seen that there were small differences in the values. With the utilisation of only 2048 samples, μ_{42}^f was much smaller for MASK. Since signals with no channel effects were used, the results are not representative for all use cases. Signal lengths under different channel conditions were thus also analysed and will be discussed in section 4.2.2.5.

The transition effects were compensated for to some extent and are explained in section 4.2.1.1. Table 5 shows the feature values without the compensation for the transition effects.

Table 5: Calculated Feature Values of Noise-free Signals over 2048 Samples without Compensation for Transitions Effects

Modulation Type	A_{mean}	σ_{aa}	σ_a	μ_{42}^a	σ_{ap}	σ_{dp}	σ_{af}	μ_{42}^f
2ASK	0.6065	0.1795	0.6343	1.3375	0.1663	0.1674	0.0314	145.4208
4ASK	0.3915	0.2190	0.4495	2.0420	0.1667	0.1677	0.0313	146.5168
2PSK	0.1174	0.1841	0.2187	14.2361	0.3511	1.5293	0.0212	118.2436
4PSK	0.1059	0.1645	0.1958	17.8525	1.0306	1.8570	0.0274	132.5924
2FSK	0.0438	0.1325	0.1410	45.9698	0.9093	1.8065	0.0119	35.4323
4FSK	0.0592	0.1490	0.1604	35.2507	0.9156	1.8080	0.0209	38.1050

It can be seen that σ_{aa} and σ_a of MPSK and MFSK were much larger and very close to the values of MASK. The same was evident for σ_{af} of MASK and MPSK. Without the removal of the fluctuations in the instantaneous amplitude and frequency, the feature values deviate significantly from the theoretical case, and become indistinguishable between many of the modulation types, which can lead to more misclassifications.

4.2.2.3 Varying SNR conditions

The first channel condition to be investigated was the effect of SNR on the feature values. For this experiment, AWGN was added to the signals. The SNR was increased by increments of 5 dB from 0 dB to 30 dB to represent very poor to fairly good noise conditions. Five hundred iterations for each SNR value were run and the averages of the five hundred iterations were calculated and plotted in Figure 45 to Figure 52. For each iteration, a new message signal and a new noise signal were generated. The feature values were calculated over a signal length of 0.12 s.

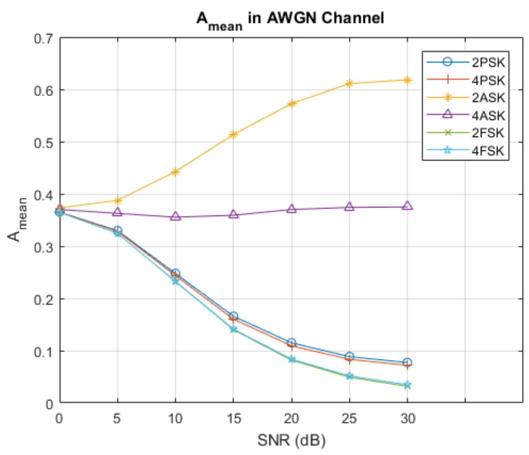


Figure 45: Feature values of A_{mean} in an AWGN Channel in Matlab Simulation

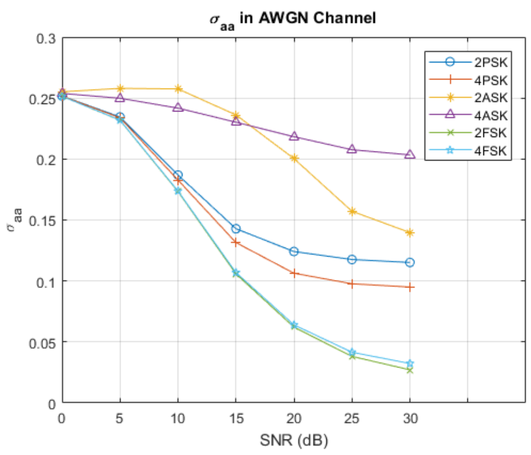


Figure 46: Feature values of σ_{aa} in an AWGN Channel in Matlab Simulation

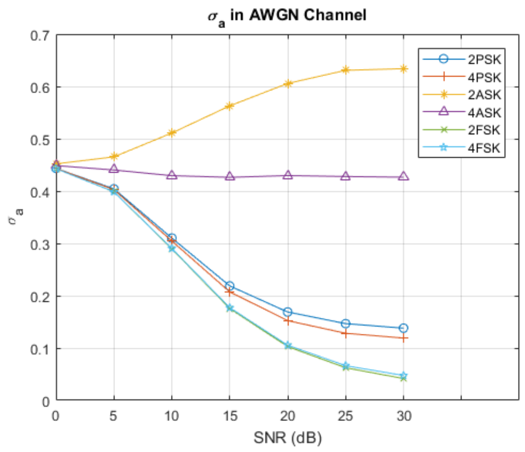


Figure 47: Feature values of σ_a in an AWGN Channel in Matlab Simulation

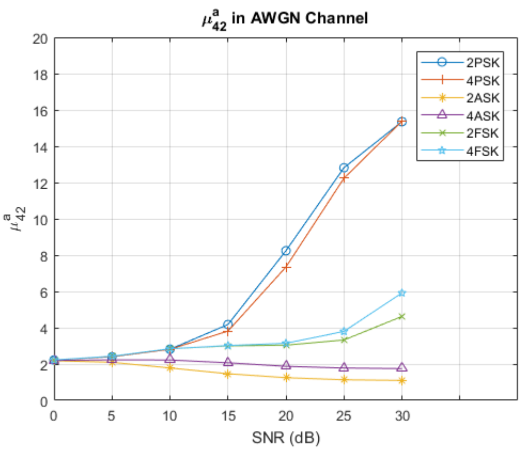


Figure 48: Feature values of μ_{42}^a in an AWGN Channel in Matlab Simulation

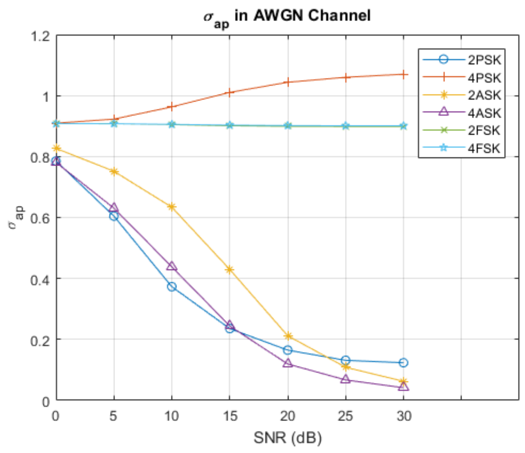


Figure 49: Feature values of σ_{ap} in an AWGN Channel in Matlab Simulation

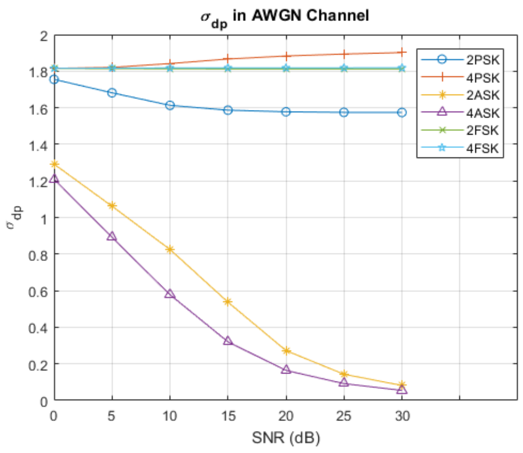


Figure 50: Feature values of σ_{dp} in an AWGN Channel in Matlab Simulation

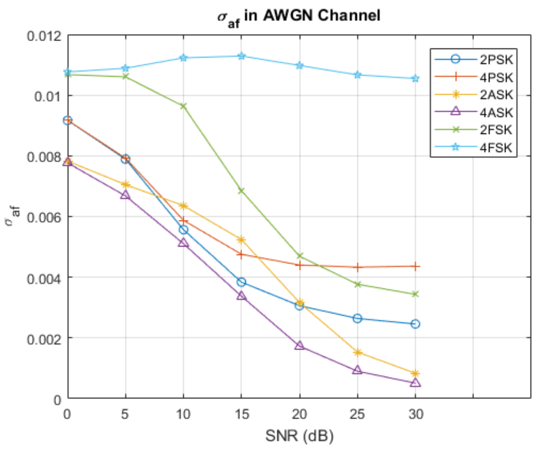


Figure 51: Feature values of σ_{af} in an AWGN Channel in Matlab Simulation

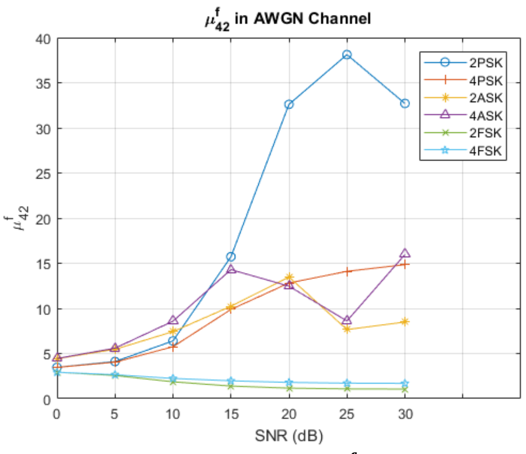


Figure 52: Feature values of μ_{42}^f in an AWGN Channel in Matlab Simulation

From these plots the following observations can be made:

For A_{mean}

- 4ASK was the least affected by the noise and the values remained almost constant over all SNR conditions. The values of MPSK as well as MFSK lied very close to each other and were not easily separable. 2ASK and 4ASK were however well separable from an SNR of 5 dB and better. MASK were also well separable from MPSK and MFSK from an SNR of 5 dB. None of the values were separable at an SNR of 0 dB.

For σ_{aa}

- 4ASK changed the least among all the modulation types. The values of MFSK lied very close to each other and were not really separable. 2ASK and 4ASK intersected at an SNR just above 15 dB which can lead to ambiguous interpretations for classification above and below this SNR. MPSK and MFSK were separable from an SNR of 10 dB and better. 2PSK and 4PSK were only separable adequately from an SNR of 15 dB. MASK were well separable from MPSK and MFSK from an SNR of 10 dB. The values of 2ASK however became closer to 2PSK as the SNR increased. None of the values were separable at an SNR of 0 dB.

For σ_a

- The results of σ_a were very similar to A_{mean}. The values were slightly larger and 2PSK and 4PSK were more separable. The values of 2PSK and 4PSK were however still not easily separable.

For μ_{42}^a

- MASK were the least affected by noise and the values remained almost constant. MPSK were the most affected by the noise and the values decreased drastically as the noise increased. MPSK and MFSK were separable from an SNR of 5 dB. MASK were also separable from MPSK and MFSK from 10 dB. None of the values were separable at an SNR of 0 dB.

For σ_{ap}

- MFSK were not affected at all by noise and remained constant over the range of SNR's from 0 dB to 30 dB. The values of MASK and 2PSK increased as the SNR decreased. 2PSK and 4ASK intersected at an SNR of 15 dB and 2PSK and 2ASK intersected at an SNR just below 25 dB. It led to ambiguous interpretations for classification above and below these SNR values. MFSK and 4PSK were however well separable from MASK and 2PSK over the range of SNR's from 0 dB to 30 dB.

For σ_{dp}

- MFSK were not affected by noise and remained constant over the range of SNR's from 0 dB to 30 dB. MASK were the most affected by the noise and increases as the SNR decreases. MASK were well separable from MPSK and MFSK over the range of SNR's from 0 dB to 30 dB. 2PSK was also separable from MFSK and 4PSK over the range of SNR's from 0 dB to 30 dB. 4PSK was separable from MFSK from 10 dB.

For σ_{af}

- 4FSK was the least affected by noise. 4FSK were also well separable from the other modulation types over the range of SNR's from 0 dB to 30 dB. 4FSK was however only separable from 2FSK from an SNR of 5 dB. 2FSK was separable from the other modulation types from 0 dB to 20 dB. Although 4ASK was close to the values of the other modulation types, it was separable from the other modulation types from an SNR of 5 dB.

For μ_{42}^{f}

- MFSK were the least affected by the noise and remained almost constant over the range of SNR's from 0 dB to 30 dB. MFSK was separable from the MASK and MPSK from an SNR of 5 dB. 2PSK was the most affected by noise and the values decreased drastically as the SNR decreased. 2PSK was separable from an SNR of 15 dB. Although 4PSK changed only a little over the SNR range, MASK caused 4PSK to be inseparable from any modulation types.

In summary, A_{mean} and σ_a had very similar results although the values differed. There was a correspondence between A_{mean} and σ_a and μ_{42}^{a}. The phase based features, σ_{ap} and σ_{dp}, of MFSK and 4PSK were the most robust against noise conditions. 4ASK was also more robust against noise when considering the amplitude based features. Similarly, 4FSK was more robust against noise when considering the frequency based features. Table 6 gives a summary of the modulation types that are distinguishable with the various features. The table is displayed to get an intuition on separability using these features in order to see if there is potential for a decision tree to work reliably on these features. The standard deviations of the features are also presented in Appendix A.1.1 to provide visual aid.

Table 6: Separability of Modulation Types under varying SNR conditions

Feature	Distinguish between	
A_{mean}	MASK vs. MPSK and MFSK	2ASK vs. 4ASK
σ_{aa}	MASK vs. MPSK and MFSK	
σ_a	MASK vs. MPSK and MFSK	2ASK vs. 4ASK
μ_{42}^{a}	MPSK vs. MFSK	MASK vs. MPSK vs. MFSK
σ_{ap}	MFSK and 4PSK vs. MASK and 2PSK	4PSK vs. MFSK
σ_{dp}	MASK vs. MPSK and MFSK	2PSK vs. rest
σ_{af}	4FSK vs. rest	2FSK vs. 4FSK
μ_{42}^{f}	MFSK vs. MASK and MPSK	

4.2.2.4 Varying SNR and flat fading conditions

The next experiment was performed to evaluate the features under varying flat fading and SNR conditions. The Matlab *RayleighChannel* System object from the Communication System Toolbox was used to create the fading channel. The average path gains were set using *AvgPathGaindB* and the path gains were then randomly generated from an internal probability distribution function for each iteration [114], [123]. A channel with 3 paths was created for each iteration. The path gains were chosen as constants and the path delays as independent variables. The path delays were chosen such that the delay span was $k \times 10\ \mu s$ where k is a calculated weight factor and was chosen as $k = [0\ \ 0.025\ \ 0.1\ \ 0.75\ \ 1.5\ \ 2.5\ \ 5\ \ 7.5\ \ 10]$. These values of k resulted in 9 different values of delay spread: [0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4]* T_S, where T_S is the symbol time. The first and last path delays were fixed at 0 μs and 10k μs respectively to ensure a delay span from 0 μs to $k \times 10$ μs in order to obtain the desired delay spread. The second path delay was chosen randomly between 2k μs and 8k μs in order to randomise the path delay of the second path for each iteration. The Doppler shift was set to 0 to create a static channel. The features were tested against these 9 delay spread values for an SNR of 10 dB and 30 dB respectively. An SNR of 30 dB is a strong signal case and from the previous analysis, at an SNR of 10 dB the features values showed good separability. Below 10 dB separability started to become an issue in some instances. Five hundred iterations for each delay spread value were performed. The averages of the five hundred iterations were calculated and plotted. For each iteration, a new message signal and noise were generated. The values were calculated over a signal length of 0.12 s. The results are shown in Figure 53 to Figure 68 where R_{DS} is the ratio of the delay spread to the symbol time. The features for 30 dB SNR are shown on the left and for 10 dB SNR on the right.

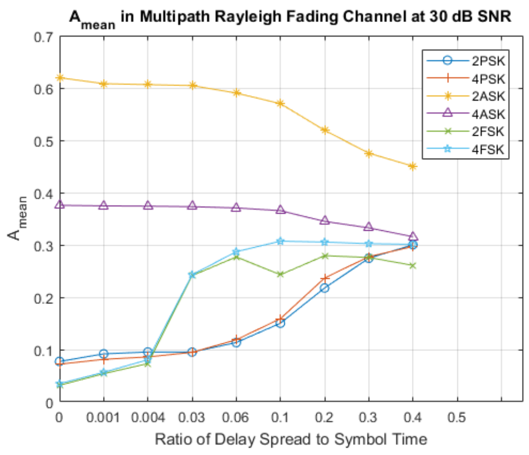


Figure 53: Feature values of A_{mean} in a Flat Fading Channel at 30 dB SNR in Matlab Simulation

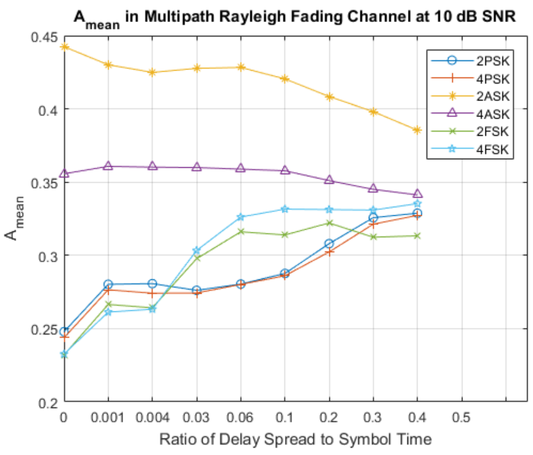


Figure 54: Feature values of A_{mean} in a Flat Fading Channel at 10 dB SNR in Matlab Simulation

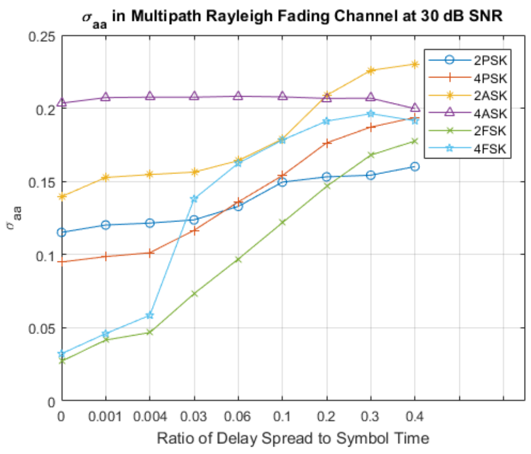


Figure 55: Feature values of σ_{aa} in a Flat Fading Channel at 30 dB SNR in Matlab Simulation

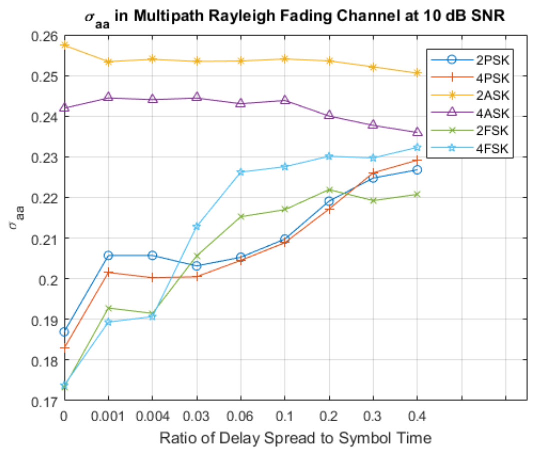


Figure 56: Feature values of σ_{aa} in a Flat Fading Channel at 10 dB SNR in Matlab Simulation

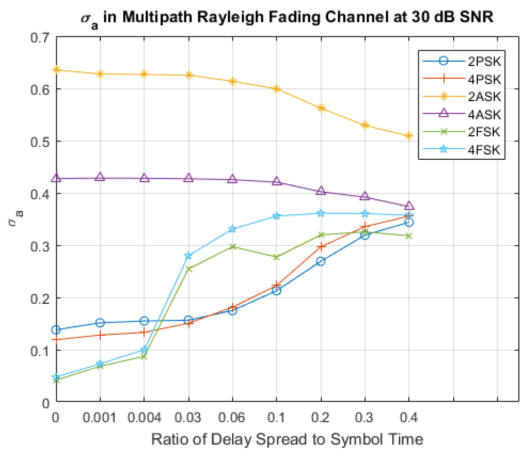


Figure 57: Feature values of σ_a in a Flat Fading Channel at 30 dB SNR in Matlab Simulation

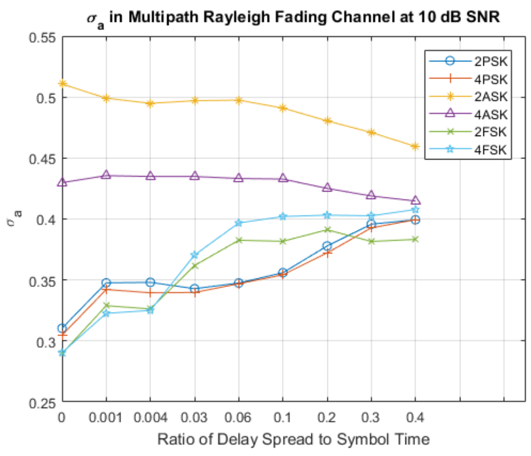


Figure 58: Feature values of σ_a in a Flat Fading Channel at 10 dB SNR in Matlab Simulation

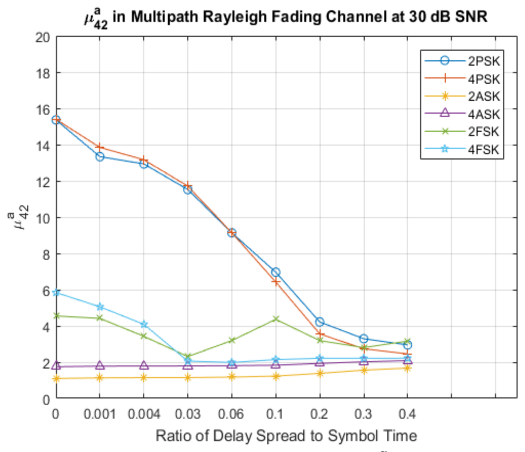


Figure 59: Feature values of μ_{42}^a in a Flat Fading Channel at 30 dB SNR in Matlab Simulation

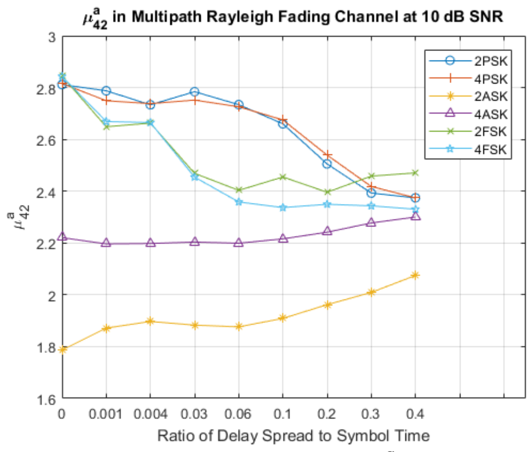


Figure 60: Feature values of μ_{42}^a in a Flat Fading Channel at 10 dB SNR in Matlab Simulation

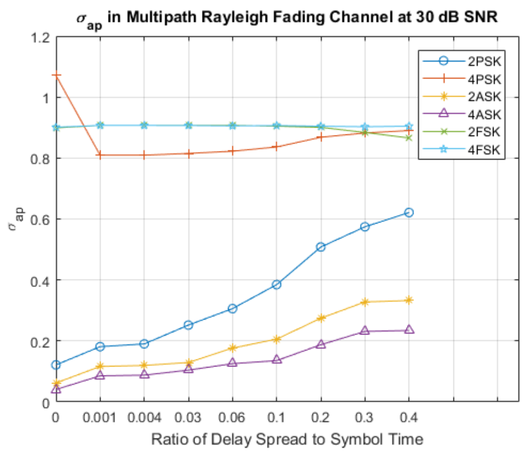


Figure 61: Feature values of σ_{ap} in a Flat Fading Channel at 30 dB SNR in Matlab Simulation

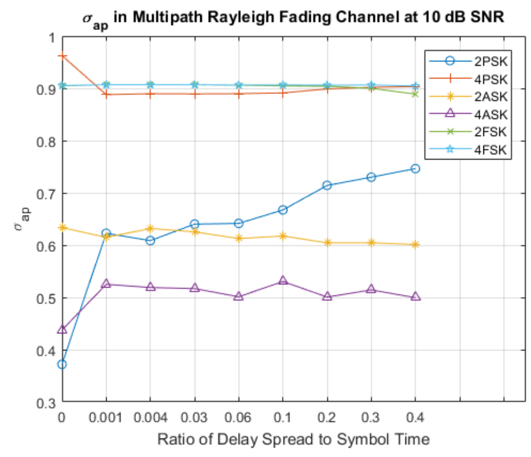


Figure 62: Feature values of σ_{ap} in a Flat Fading Channel at 10 dB SNR in Matlab Simulation

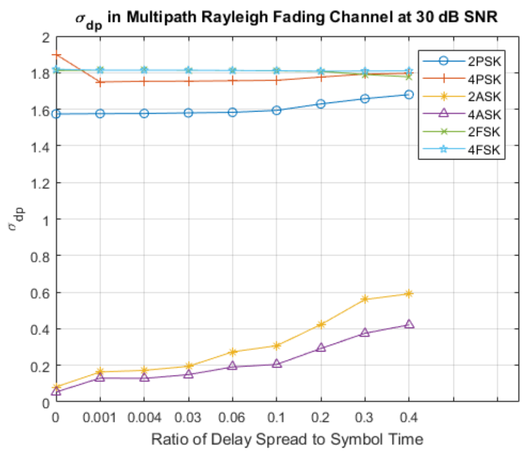


Figure 63: Feature values of σ_{dp} in a Flat Fading Channel at 30 dB SNR in Matlab Simulation

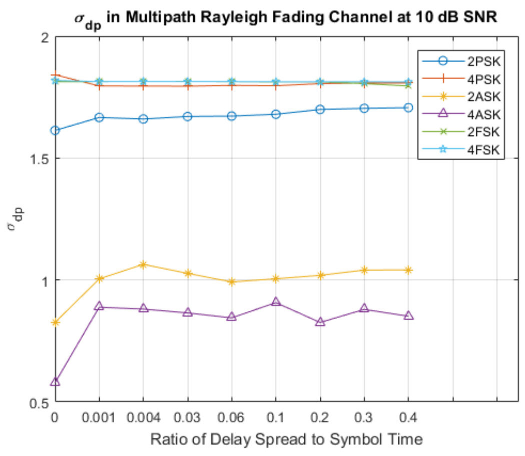


Figure 64: Feature values of σ_{dp} in a Flat Fading Channel at 10 dB SNR in Matlab Simulation

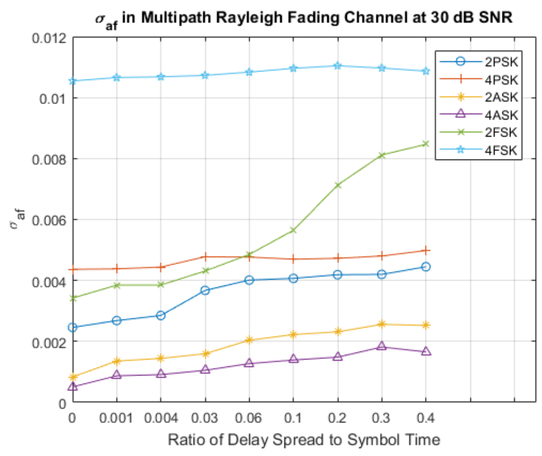


Figure 65: Feature values of σ_{af} in a Flat Fading Channel at 30 dB SNR in Matlab Simulation

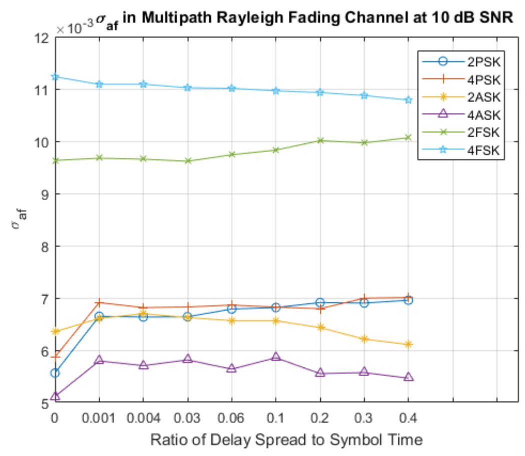


Figure 66: Feature values of σ_{af} in a Flat Fading Channel at 10 dB SNR in Matlab Simulation

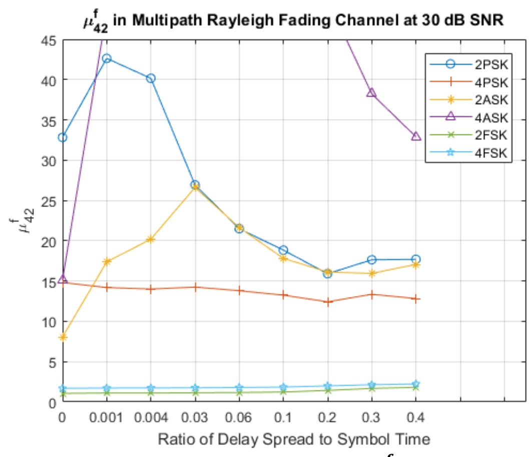


Figure 67: Feature values of μ^f_{42} in a Flat Fading Channel at 30 dB SNR in Matlab Simulation

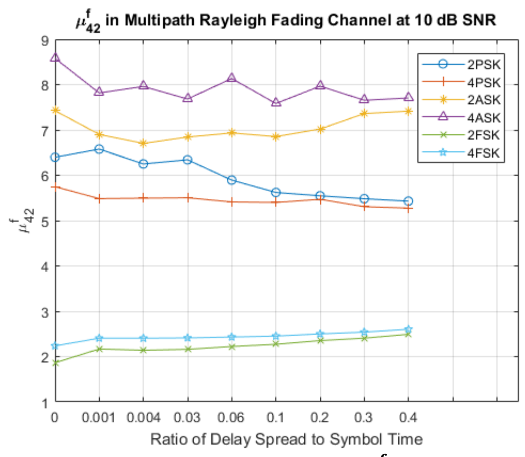


Figure 68: Feature values of μ^f_{42} in a Flat Fading Channel at 10 dB SNR in Matlab Simulation

The results show that:

For A_{mean}

- 4ASK was the least affected by the flat fading and remained almost constant for all R_{DS} from 0 to 0.4 for both SNR's of 30 dB and 10 dB. MASK were well separable from MPSK and MFSK for all R_{DS} from 0 to 0.4 for both SNR values. 4ASK was however not separable for R_{DS} of 0.4. 2ASK was also well separable from 4ASK over all the proposed channel conditions.

For σ_{aa}

- 4ASK was the least affected by the flat fading and remained almost constant for all R_{DS} from 0 to 0.4 for SNR of 30 dB. 4ASK was also separable from the other modulation types for all R_{DS} from 0 to 0.4 for an SNR of 10 dB and for R_{DS} from 0 to 0.1 for an SNR of 30 dB. The effect of noise on 2ASK can be seen here. The feature values give ambiguous results if the SNR of signals are unknown. Modulation types were not separable in the same way at an SNR of 30 dB and 10 dB. MASK was however separable from MPSK and MFSK for all R_{DS} from 0 to 0.4 for an SNR of 10 dB.

For σ_a

- MASK was the least affected by the flat fading conditions. 2ASK remained separable from all the other modulation types for all R_{DS} from 0 to 0.4 for both SNR values. 4ASK were also separable for all R_{DS} from 0 to 0.4 for both SNR values. The value of 4ASK was however very close to 4FSK at an R_{DS} of 0.4. The values of 2PSK and 4PSK were also very close to each other from all R_{DS} from 0 to 0.4 for both SNR values.

For μ_{42}^{a}

- MASK were the least affected by the flat fading and remained almost constant for all R_{DS} from 0 to 0.4 for an SNR of 30 dB. MPSK were the most affected by both SNR and flat fading. The values decreased drastically as R_{DS} increased. 2ASK was separable from the other modulation types for all R_{DS} from 0 to 0.4 for both SNR values. 4ASK was separable from the other modulation types for all R_{DS} from 0 to 0.4 for an SNR of 10 dB. The value of 4ASK was however very close to 4FSK at an R_{DS} of 0.4. Although 4ASK remained almost constant at an SNR of 30 dB, 4FSK caused 4ASK to only be separable for R_{DS} from 0 to 0.03.

For σ_{ap}

- MFSK were the least affected by the flat fading conditions and remained almost constant for all R_{DS} from 0 to 0.4 for both SNR values. MFSK and 4PSK remained well separable from MASK and 2PSK for all R_{DS} from 0 to 0.4 for both SNR values although 2ASK was greatly affected by noise.

For σ_{dp}

- MFSK and 4PSK were the least affected by the flat fading and remained almost constant for all R_{DS} from 0 to 0.4 for an SNR of 30 dB. Although MASK were greatly affected by noise, MASK were still very well separable from MPSK and MFSK for all R_{DS} from 0 to 0.4 for both SNR values. MFSK were also separable from 4PSK for all the proposed channel conditions.

For σ_{af}

- 4FSK was the least affected by the flat fading conditions. 4FSK was well separable for all R_{DS} from 0 to 0.4 for both SNR values. 2FSK was separable for all R_{DS} from 0 to 0.4 for an SNR of 10 dB. The effect of noise on 2FSK can be seen here. The feature values give ambiguous results if the SNR of signals are unknown. Modulation types are not separable in the same way at an SNR of 30 dB and 10 dB. 2ASK and 4ASK were however separable for all the proposed channel conditions. MPSK were also well separable from MASK for all R_{DS} from 0 to 0.4 for an SNR of 30 dB.

For μ_{42}^{f}

- MFSK were the least affected by the flat fading conditions and were well separable from MASK and MPSK for all R_{DS} from 0 to 0.4 for both SNR values. MASK were also separable from MPSK for all R_{DS} from 0 to 0.4 for an SNR of 10 dB. 4ASK was greatly affected by flat fading. The effect on MASK and 2PSK can lead to ambiguous interpretations for classification.

A_{mean} and σ_a again showed very similar trends although the values differed. It can also be observed that there was a correspondence between A_{mean} and σ_a, and μ_{42}^{a}. It can be seen that σ_{ap} and σ_{dp} of MFSK and 4PSK were more robust against flat fading conditions. The amplitude based features of MASK were also more robust against flat fading as well as the frequency based features of 4FSK. Table 7 gives a summary of the modulation types that are distinguishable with the various features. The standard deviations of the features are also presented in Appendix A.1.2 to provide visual aid for separability of the feature values. It can be seen that the features were more sensitive to fading conditions and misclassification might occur more.

Table 7: Separability of Modulation Types under varying flat fading and SNR conditions

Feature	Distinguish between	
A_{mean}	MASK vs. MPSK and MFSK	2ASK vs. 4ASK
σ_{aa}	MASK vs. MPSK and MFSK	4ASK vs. MPSK and MFSK
σ_a	MASK vs. MPSK and MFSK	2ASK vs. 4ASK
μ_{42}^{a}	MPSK vs. MFSK	
σ_{ap}	MFSK and 4PSK vs. MASK and 2PSK	2ASK vs. 4ASK
σ_{dp}	MASK vs. MPSK and MFSK	2PSK vs. rest
σ_{af}	4FSK vs. rest	2ASK vs. 4ASK
μ_{42}^{f}	MFSK vs. MASK and MPSK	

4.2.2.5 *Signal Length Effects*

Shorter signals consume less hardware resources and reduce calculation time. Additionally, since transmitters in a non-cooperative environment can change their modulation type quickly, it is also desirable to classify modulation types using the fewest samples possible for quicker classification turnaround time. The aim of the experiment is to determine how sensitive the features were to signal length. This is to aid in the decision for minimum signal length to still obtain good performance. The same signals generated in section 4.2.2.4 were used for this experiment. One hundred signals for each fading value were used for an SNR of 10 dB and 30 dB respectively. The feature values were calculated for 21 different signal lengths. For the maximum signal length, signals of 120 ms were used. This is equal to:

$$\begin{aligned} \#\, of\ samples &= signal\ time\ \times\ f_s \\ &= (120\ \times 10^{-3})(1200 \times 10^{3}) \\ &= 144\ \times 10^{3}\ samples \end{aligned} \tag{52}$$

The signal length is decreased by $\frac{144\ \times 10^{3}}{15i}$ where $i = 1,2,3, \ldots, 20$. The last iteration thus consists of 480 samples. For a symbol rate of 12.5 $kbaud$ and a sample frequency of 1200 kHz, each symbol consists of 96 samples. The last iteration is thus calculated over 5 symbols. The averages of the iterations were calculated and the maximum and minimum values among all the fading conditions were plotted for each signal length. The results are shown in Figure 69 to Figure 84. Each figure shows the maximum and minimum feature value over all the fading conditions for each modulation type which are indicated as "max" and "min" on the graphs.

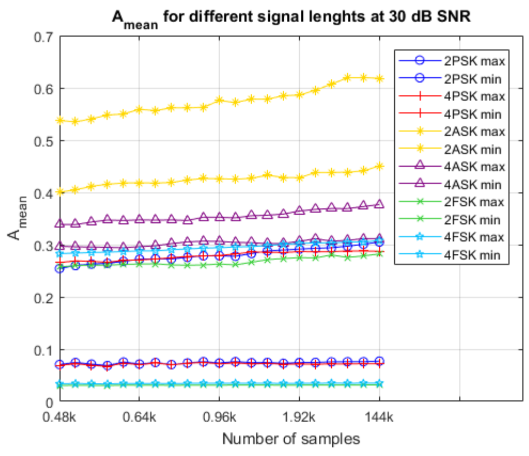


Figure 69: Minimum and Maximum Feature values of A_{mean} in a Flat Fading Channel for different Signal Lengths at 30 dB SNR

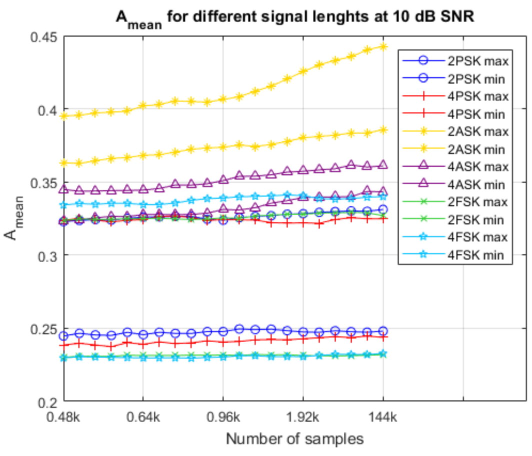


Figure 70: Minimum and Maximum Feature values of A_{mean} in a Flat Fading Channel for different Signal Lengths at 10 dB SNR

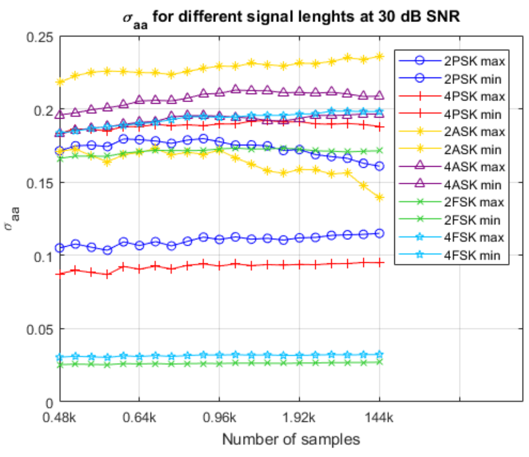


Figure 71: Minimum and Maximum Feature values of σ_{aa} in a Flat Fading Channel for Different Signal Lengths at 30 dB SNR

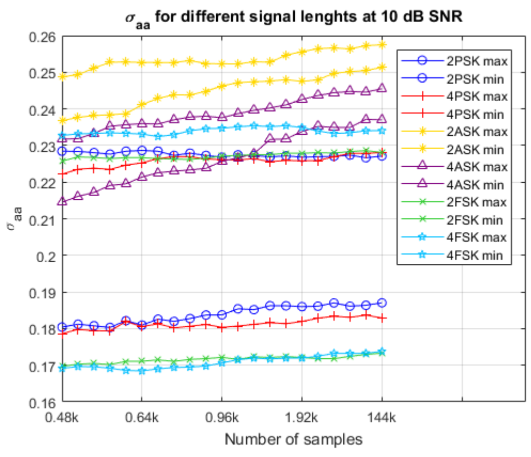


Figure 72: Minimum and Maximum Feature values of σ_{aa} in a Flat Fading Channel for Different Signal Lengths at 10 dB SNR

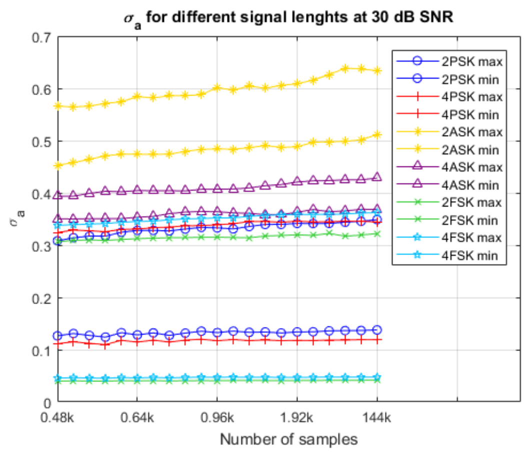


Figure 73: Minimum and Maximum Feature values of σ_a in a Flat Fading Channel for Different Signal Lengths at 30 dB SNR

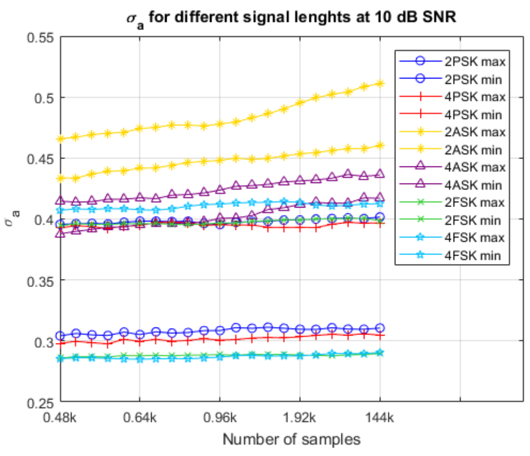


Figure 74: Minimum and Maximum Feature values of σ_a in a Flat Fading Channel for Different Signal Lengths at 10 dB SNR

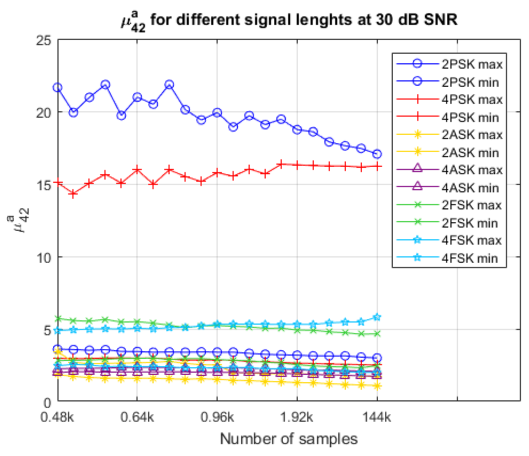


Figure 75: Minimum and Maximum Feature values of μ_{42}^a in a Flat Fading Channel for Different Signal Lengths at 30 dB SNR

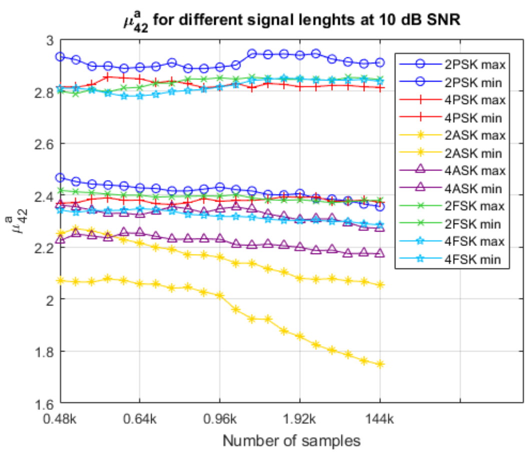


Figure 76: Minimum and Maximum Feature values of μ_{42}^a in a Flat Fading Channel for Different Signal Lengths at 10 dB SNR

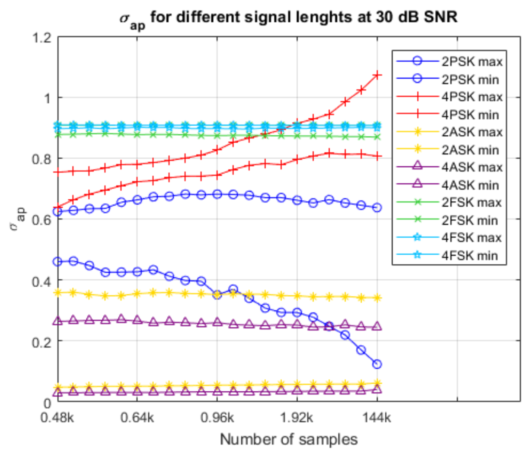


Figure 77: Minimum and Maximum Feature values of σ_{ap} in a Flat Fading Channel for Different Signal Lengths at 30 dB SNR

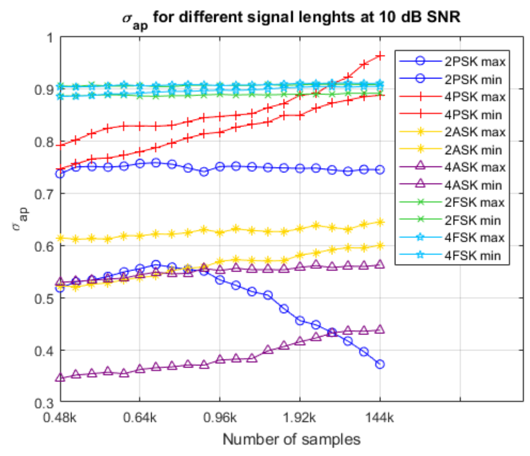


Figure 78: Minimum and Maximum Feature values of σ_{ap} in a Flat Fading Channel for Different Signal Lengths at 10 dB SNR

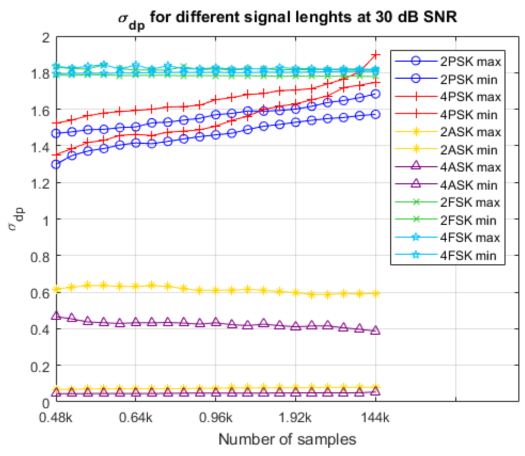


Figure 79: Minimum and Maximum Feature values of σ_{dp} in a Flat Fading Channel for Different Signal Lengths at 30 dB SNR

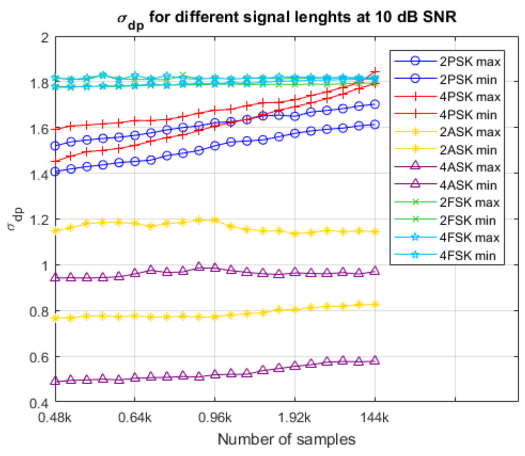


Figure 80: Minimum and Maximum Feature values of σ_{dp} in a Flat Fading Channel for Different Signal Lengths at 10 dB SNR

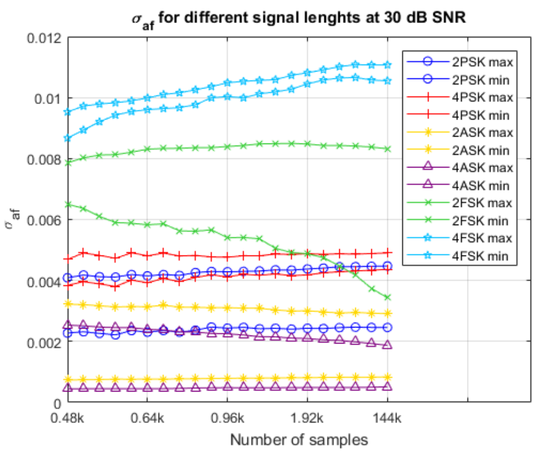


Figure 81: Minimum and Maximum Feature values of σ_{af} in a Flat Fading Channel for Different Signal Lengths at 30 dB SNR

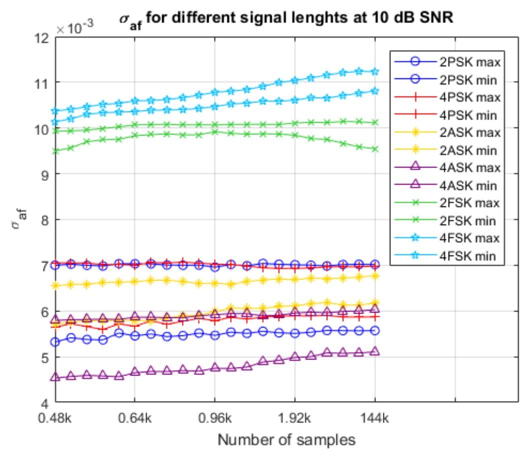


Figure 82: Minimum and Maximum Feature values of σ_{af} in a Flat Fading Channel for Different Signal Lengths at 10 dB SNR

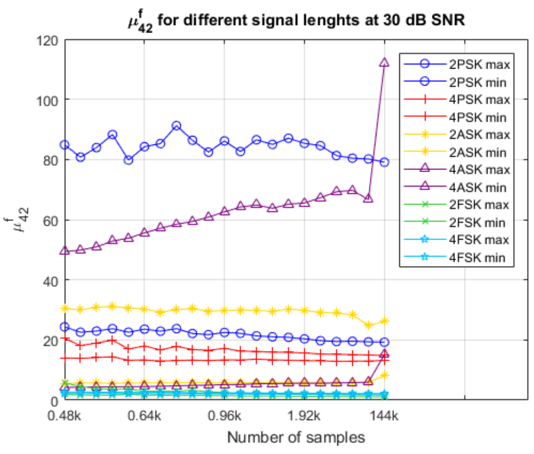


Figure 83: Minimum and Maximum Feature values of μ_{42}^{f} in a Flat Fading Channel for Different Signal Lengths

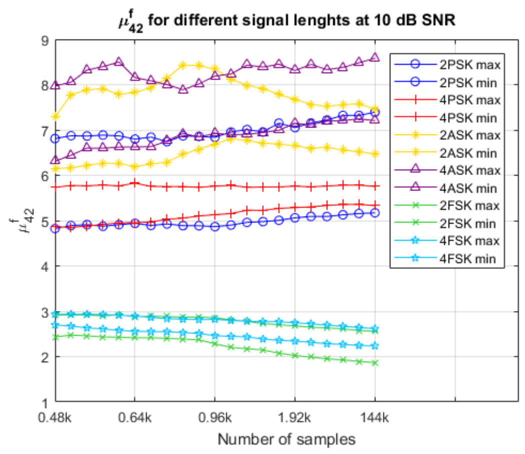


Figure 84: Minimum and Maximum Feature values of μ_{42}^{f} in a Flat Fading Channel for Different Signal Lengths

The results show that:

For A_{mean}:

- At 30 dB SNR most modulation types remained relatively constant over the extent of the proposed signal lengths. Although the maximum values of MPSK only changed a little, MPSK intersected MFSK at the maximum values and can lead to ambiguous results when fewer signal samples are used. Although there was a decrease in the values of 2ASK, it remained separable from the other modulation types. The minimum values of 2PSK and 4PSK as well as the minimum values of 2FSK and 4FSK were inseparable over the whole range of signal lengths.
- At 10 dB SNR most modulation types also remained relatively constant over the extent of the proposed signal lengths. The maximum values of 2PSK became inseparable from the maximum values of 2FSK and 4PSK. The minimum values of 2FSK and 4FSK were inseparable over the whole range of signal lengths.

For σ_{aa}

- At 30 dB SNR MFSK and MPSK remained relatively constant over the extent of the proposed signal lengths. The maximum values of 2PSK however increased as the signal length decreased and intersect 2FSK at the maximum values. The minimum value of 2ASK changed the most over the decreasing signal lengths and also intersected the maximum values of 2FSK. The minimum values of 4ASK and the maximum values of 4FSK became inseparable. It can also be noted that the minimum and maximum values of MFSK were distributed far apart, showing the negative effects of fading on this feature.
- At 10 dB SNR MASK decreased as the signal length decreased. The minimum values of 4ASK intersected the maximum values of all the other modulation types and can lead to ambiguous results when fewer signal samples are used. Although 2ASK decreased, it remained separable from the other modulation types.

For σ_a

- At 30 dB SNR MFSK and MPSK remained relatively constant over the extent of the proposed signal lengths. The maximum values of MPSK and MFSK as well as the minimum values of 4ASK were very close which could lead to misclassification. This observation was however evident for all the signal lengths and not only for shorter signal lengths. Both maximum and minimum values of 2ASK remained separable from the other modulation types for all signal lengths.
- At 10 dB SNR similar results were obtained. The minimum values of 4ASK however intersected the maximum values of MPSK and MFSK which could lead to ambiguous results when fewer signal samples are used.

For μ_{42}^{a}

- At 30 dB the maximum values of MPSK varied over the range of signal lengths, but remained separable. Although the minimum values of all the modulation types remained almost constant for all the proposed signal lengths, the values were very close. This observation was however evident for all the signal lengths and not only for shorter signal lengths. It can be seen that the maximum values of 2FSK and 4FSK intersected due to the fact that 2FSK decreased and 4FSK increased as the signal length decreased.

- At 10 dB SNR the maximum and minimum values of all the modulation types, except for 2ASK showed similar results. The values were very close to each other and multiple intersections occurred, which may lead to ambiguous results. 2ASK increased notably as the signal length decreased and the maximum values eventually intersected the minimum values of 4ASK.

For σ_{ap}

- At 30 dB SNR MPSK were greatly affected by signal length. 4PSK decreased and 2PSK increased as the signal length decreased. This caused the maximum values of 4PSK to intersect the values of MFSK and the minimum values of 2PSK to intersect the maximum values of MASK. This effect could lead to ambiguous results if different signal lengths are used for classification. MASK and MFSK remained almost constant for all the proposed signal lengths.
- At 10 dB SNR similar results were obtained. Since 2ASK had different values at 10 dB SNR, the minimum values of 2PSK intersected the minimum values of 2ASK. The minimum values of 2ASK also intersected the maximum values of 4ASK at shorter signal lengths.

For σ_{dp}

- At 30 dB SNR MASK and MFSK remained relatively constant over the extent of the proposed signal lengths. MPSK were the most affected by signal length. MASK, MPSK and MFSK were separable from each other for the proposed signal lengths. The maximum values of 4PSK however intersected the values of MFSK for longer signal lengths. 2PSK and 4PSK intersected for shorter signal lengths. Most of the values of 4ASK were within the range of 2ASK for all proposed signal lengths. MASK remained well separable from MPSK and MFSK for all signal lengths.
- At 10 dB SNR similar results were obtained. 2ASK however had larger values than at 30 dB, which shows the effect of SNR on this feature.

For σ_{af}

- At 30 dB MASK and MPSK remained almost constant for all the proposed signal lengths. The minimum values of 2FSK were the most affected by signal length. The values increased as signal length decreased. 2FSK and MPSK intersected and can lead to ambiguous results when different signal lengths are used for classification. It can however be noted that the distribution between the maximum and minimum values became smaller. Although the maximum and minimum values of 4FSK decreased as signal length decreases, 4FSK remained well separable from the other modulation types.
- At 10 dB similar results were obtained. The distribution of the minimum and maximum values of 2FSK were however smaller and the minimum values did not intersect with any other modulation types.

For μ_{42}^{f}

- At 30 dB SNR MPSK, MFSK and 2ASK remained almost constant for all the proposed signal lengths. 4ASK showed a significant decrease for the maximum values with the decrease of signal length and the maximum values of 4ASK were under the maximum values of 2PSK. It may lead to ambiguous results when different signal lengths are used for classification. The minimum values of 2ASK and 4ASK became inseparable.
- At 10 dB SNR MPSK and MFSK remained almost constant over the range of signal lengths. The values of MASK varied and caused intersections at various signal lengths. It may lead to ambiguous results when different signal lengths are used for classification.

4.2.2.6 Decision tree

The feature values obtained from the different experiments of section 4.2.2.3 and 4.2.2.4 were used for training and testing of the decision tree. The training dataset consisted of four hundred training vectors of each SNR condition from 0 dB to 30 dB for each of the six modulation types. Furthermore, the training dataset consisted of four hundred training vectors of each fading condition at 30 dB and 10 dB SNR respectively for each of the six modulation types. The decision tree used these values to grow the tree and used 10-fold cross validation to prune the tree. The following number of training vectors was used from each dataset:

- SNR : 400 training vectors of each SNR condition (7 x 6 x 400 = 16800)
- Fading at 10 dB : 400 training vectors of each fading ratio condition (9 x 6 x 400 = 21600)
- Fading at 30 dB : 400 training vectors of each fading ratio condition (9 x 6 x 400 = 21600)

Due to the sizes of the trees, they are not presented here diagrammatically. The results of classification after training of the tree on the above dataset are shown in Table 8.

Table 8: Results of the construction of the decision tree

		Before Pruning			After Pruning			
Test	Training vectors	Tree depth (levels)	Re-substitution Error (%)	Cross Validation Error (%)	Tree depth (levels)	Re-substitution Error (%)	Cross Validation Error (%)	Estimated Classification Error (%)
1	60 000	111	1.04	4.14	58	2.73	4.09	3.83

The test dataset consisted of one hundred test vectors of each SNR condition from 0 dB to 30 dB for each of the six modulation types. Furthermore, the test dataset consisted of one hundred test vectors of each fading condition at 30 dB and 10 dB SNR respectively for each of the six modulation types. The same test dataset was used for all the tests of the decision trees. The following number of test vectors was used from each dataset generated in the different tests for each modulation type:

- SNR : 100 test vectors of each SNR condition (7 x 6 x 100 = 4200)
- Fading at 10 dB : 100 test vectors of each fading ratio condition (9 x 6 x 100 = 5400)
- Fading at 30 dB : 100 test vectors of each fading ratio condition (9 x 6 x 100 = 5400)

Confusion matrices were calculated to evaluate the performance of the decision tree for the different datasets. The confusion matrices show the correct classification and the misclassification for each class. A confusion matrix for a two-class (Positive and Negative) classification problem can be presented as follows:

		Predicted classes	
		Positive	Negative
True classes	Positive	True Positive	False Negative
	Negative	False Positive	True Negative

Figure 85: Confusion matrix for a two-class classification problem

The true class labels are in the rows and the predicted class labels in the columns. For true positive and true negative, the classes are correctly classified. For a false positive, a negative is misclassified as a positive. For a false negative, a positive is misclassified as negative.
The confusion matrices, calculated for each SNR as well as each fading ratio at both 30 dB and 10 dB SNR, can be seen in Appendix B.1 and will be discussed below. Table 9 and Table 10 show the classification accuracy achieved by the decision tree for the proposed conditions.

Table 9: Classification accuracy (%) of the decision tree with full training dataset over varying SNR

SNR (dB)	0	5	10	15	20	25	30
Classification accuracy (%)	96.17	99.67	100	100	100	100	100

Table 10: Classification accuracy (%) of the decision tree with full training dataset over varying R_{DS} at 30 dB and 10 dB SNR

R_{DS}	0	0.001	0.004	0.03	0.06	0.1	0.2	0.3	0.4
Classification accuracy at 30 dB (%)	100	99.67	99.5	99.5	99.17	98.83	97.33	92.00	89.33
Classification accuracy at 10 dB (%)	100	95.33	97.17	94.5	97.17	91.67	91.5	86.17	84.83

From Table 9 and Table 10 and the confusion matrices in Appendix B.1 it can be seen that:

- The classification accuracy decreased as the fading conditions became worse and the SNR decreased.
- The classification accuracy of the tree was very good for the SNR dataset. The tree had 100% classification accuracy down to an SNR of 5 dB. For an SNR of 0 dB a 96.17% classification accuracy was obtained. The most misclassifications occurred between 2FSK and 4FSK.
- The performance of the tree was very good up to an R_{DS} of 0.2 for the fading dataset at an SNR of 30 dB. The classification accuracy was above 97.33%. For an R_{DS} of 0.3 the classification accuracy decreased to 92% and for an R_{DS} of 0.4 to 89.33%. It can be noted that misclassification mostly occurred between the MASK pairs, MPSK pairs and MFSK pairs and not between different classes of modulations. For an R_{DS} of 0.4 the misclassification between the MASK pairs, MPSK pairs and MFSK pairs increased.

- The classification accuracy varied for the different R_{DS} for the fading dataset at 10 dB. From an R_{DS} of 0.1 the classification accuracy started to decrease from 91.67% to 84.83% for an R_{DS} of 0.4. The performance of the tree was however better than 91% up to an R_{DS} of 0.2. The misclassification also mostly occurred between the MASK pairs, MPSK pairs and MFSK pairs. The misclassification of 2PSK as 2ASK or 4ASK was also notable from an R_{DS} of 0.3.

The following experiment was performed to investigate the training sensitivity to the dataset size. The size of the training datasets were decreased to three hundred, two hundred, one hundred, and fifty training vectors of each SNR and each fading condition respectively for each of the six modulation types.

Table 11: Results of the construction of the decision trees from five different training datasets

		Before Pruning			After Pruning			
Test	Training vectors	Tree depth (levels)	Re-substitution Error (%)	Cross Validation Error (%)	Tree depth (levels)	Re-substitution Error (%)	Cross Validation Error (%)	Estimated Classification Error (%)
1	60 000	111	1.04	4.14	58	2.73	4.09	3.83
2	45 000	88	1.09	4.30	53	2.50	4.20	3.91
3	30 000	69	1.24	4.44	37	3.08	4.51	4.26
4	15 000	53	1.11	4.77	26	3.35	4.95	4.59
5	7500	32	1.55	5.71	23	3.09	5.39	5.24

It can be seen that the complexity of the tree reduced by more than half when less samples were used, while the estimated classification error only increased a little.

For the rest of the document, these five types of datasets used for training, including a full dataset containing 400 training vectors of each channel condition, will be referred to as 400, 300, 200,100, and 50 training vectors respectively. Table 12 shows the classification accuracy of the decision trees over varying SNR's.

Table 12: Classification accuracy (%) of decision trees with decreasing training sets over varying SNR using Feature Values obtained in Software

Training vectors		**SNR (dB)**						
		0	**5**	**10**	**15**	**20**	**25**	**30**
Test1	400	96.17	100	100	100	100	100	100
Test2	300	96.17	99.67	100	100	100	100	100
Test3	200	96.17	100	100	100	100	100	100
Test4	100	94.83	100	100	100	100	100	100
Test5	50	93.17	100	100	100	100	100	100

From Table 12 it can be seen that the classification accuracy was very high and 100% classification accuracy was achieved for most tests of SNR greater than 0 dB. None of the tests achieved 100% classification accuracy for an SNR of 0 dB. Classification accuracy greater than 93% was however achieved for all the tests and a decrease in accuracy of 3% is observed between the biggest and smallest training dataset utilised.

The following table and figures show the results of the decision trees for varying flat fading conditions at 30 dB SNR.

Table 13: Classification accuracy (%) of decision trees with decreasing training sets over varying R_{DS} at 30 dB SNR using Feature Values obtained in Software

Training vectors		R_{DS}								
		0	**0.001**	**0.004**	**0.03**	**0.06**	**0.1**	**0.2**	**0.3**	**0.4**
Test1	400	100	99.67	99.5	99.5	99.17	98.83	97.33	92.00	89.33
Test2	300	100	99.83	99.33	99.5	99.5	99.00	96.83	92.33	88.83
Test3	200	100	99.67	99.50	99.33	99.67	98.83	96.83	92.33	88.00
Test4	100	100	99.5	99.17	99.00	97.83	98.17	96.50	93.67	88.17
Test5	50	100	99.67	99.50	98.50	98.33	98.00	96.67	91.17	87.00

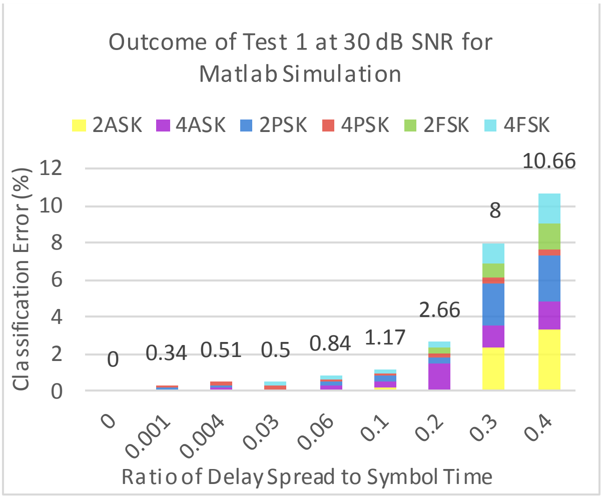


Figure 86: Classification Error for 400 training vectors under varying flat fading conditions at 30 dB SNR

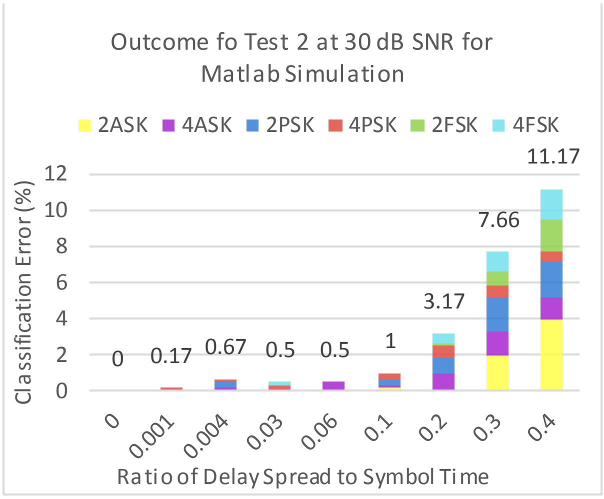


Figure 87: Classification Error for 300 training vectors under varying flat fading conditions at 30 dB SNR

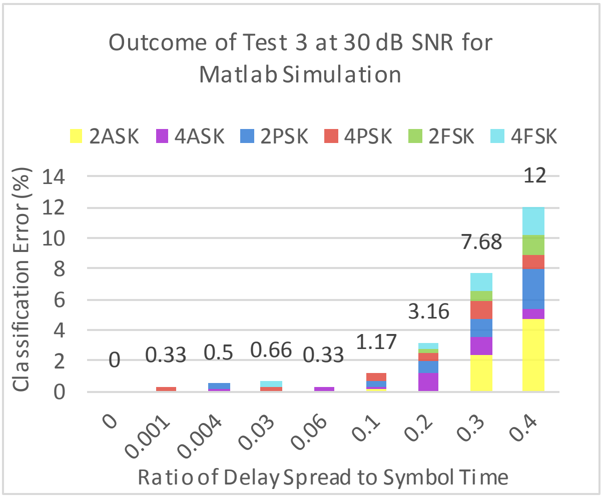


Figure 88: Classification Error for 200 training vectors under varying flat fading conditions at 30 dB SNR

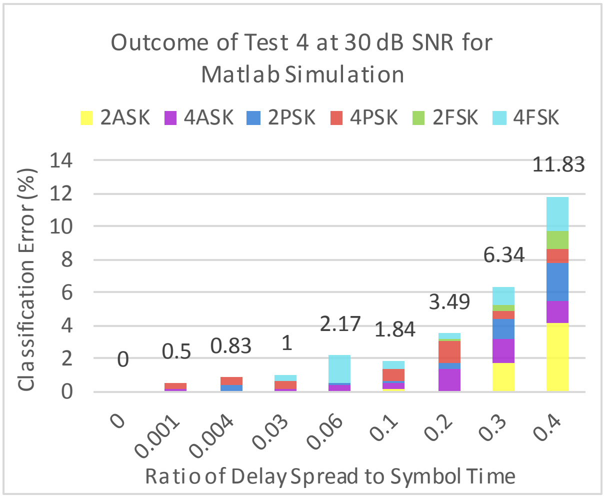


Figure 89: Classification Error for 100 training vectors under varying flat fading conditions at 30 dB SNR

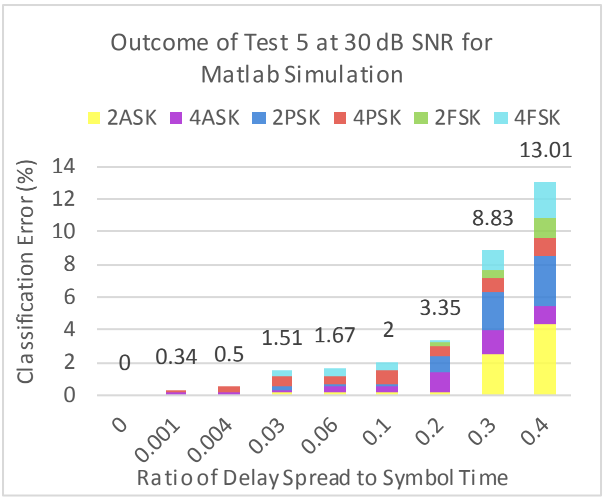


Figure 90: Classification Error for 50 training vectors under varying flat fading conditions at 30 dB SNR

From Table 13 and Figure 86 to Figure 90, it can be seen that the classification accuracy decreased as the fading conditions deteriorated. Classification accuracies higher than 91% were achieved for all tests up to a fading ratio of 0.3 and higher than 87% for a fading ratio of 0.4. The largest decrease in accuracy between the biggest and smallest training dataset utilised was smaller than 2% and occurred at a fading ratio of 0.4. The misclassification of 2ASK and 2PSK was the main contribution to the classification errors. There was a significant increase in classification error from a fading ratio of 0.2 to a fading ratio of 0.3. At fading ratios larger than 0.2 the misclassification of 2ASK started to occur. Table 14 and Figure 91 to Figure 95 show the classification accuracy of the decision tree for varying flat fading conditions at 10 dB SNR.

Table 14: Classification accuracy (%) of decision trees with decreasing training sets over varying R_{DS} at 10 dB SNR using Feature Values obtained in Software

Training vectors		R_{DS}								
		0	**0.001**	**0.004**	**0.03**	**0.06**	**0.1**	**0.2**	**0.3**	**0.4**
Test1	400	100	95.33	97.17	94.5	97.17	91.67	91.5	86.17	84.83
Test2	300	100	95.17	96.33	94.17	95.33	93.00	90.33	88.67	82.5
Test3	200	100	94.50	96.83	92.67	95.33	90.67	90.83	85.67	83.00
Test4	100	100	92.50	96.33	93.00	94.33	90.83	91.5	84.33	81.67
Test5	50	100	92.83	94.17	90.67	92.00	87.50	88.83	81.83	79.33

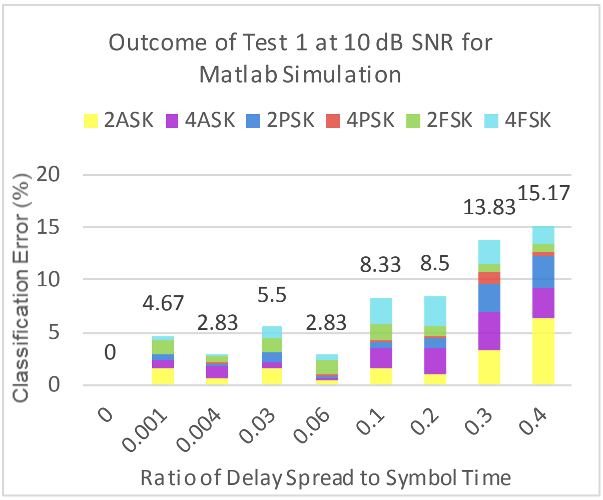


Figure 91: Classification Error for 400 training vectors under varying flat fading conditions at 10 dB SNR

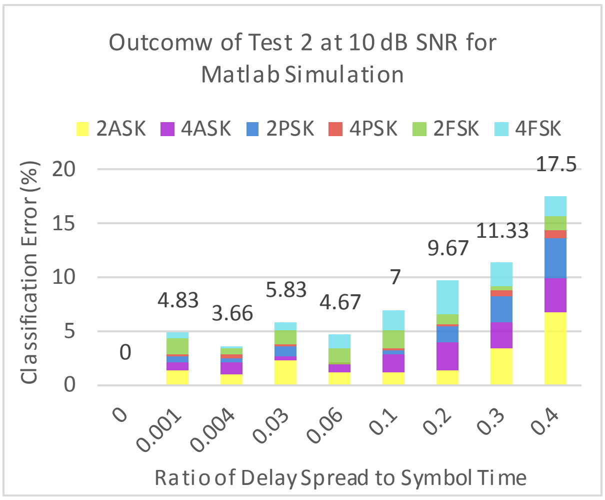


Figure 92: Classification Error for 300 training vectors under varying flat fading conditions at 10 dB SNR

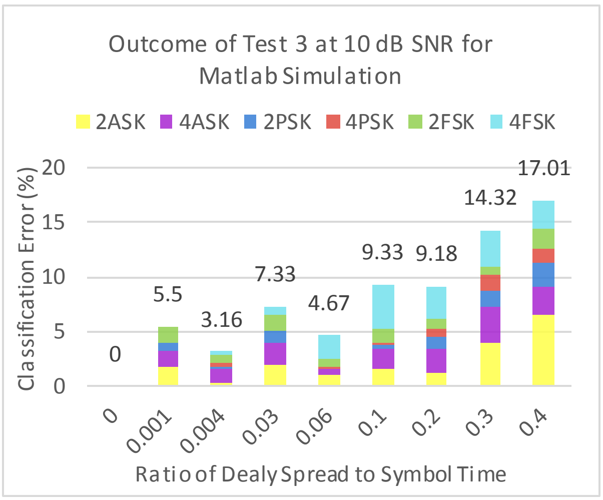


Figure 93: Classification Error for 200 training vectors under varying flat fading conditions at 10 dB SNR

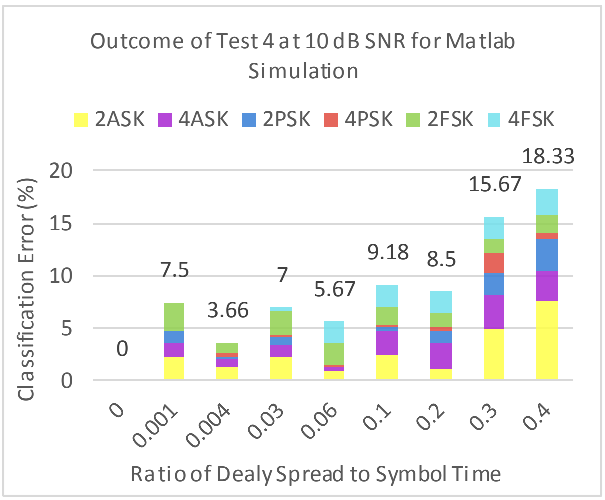


Figure 94: Classification Error for 100 training vectors under varying flat fading conditions at 10 dB SNR

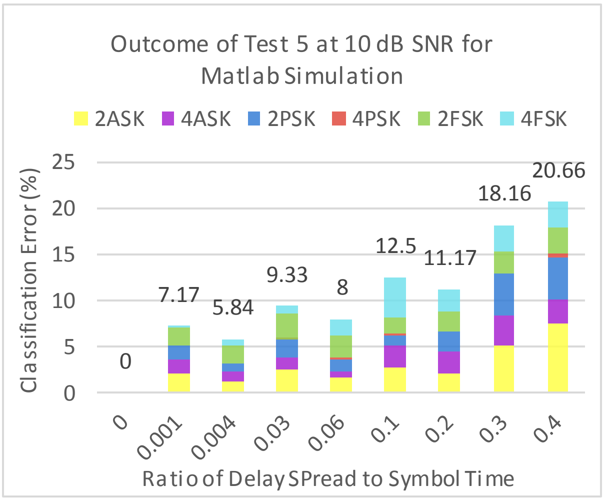


Figure 95: Classification Error for 50 training vectors under varying flat fading conditions at 10 dB SNR

From Table 14 and Figure 91 to Figure 95, it can be seen that there was a significant decrease in classification accuracy. Classification accuracies higher than 79% were achieved for all tests under all the flat fading conditions. The largest decrease in classification accuracy between the biggest and smallest training dataset utilised, was smaller than 6% and occurred at a fading ratio of 0.06. The misclassification of 2ASK and 2PSK was the main contribution to the classification errors. There was a significant increase in classification error from a fading ratio of 0.2 to a fading ratio of 0.3. At fading ratios larger than 0.2 the misclassification of 2ASK started to occur. It is evident that 4PSK had the least classification errors. It can also be seen that the classification errors were smaller for some worse fading conditions which is counter intuitive and against the general trend in the data. This can be seen for a fading ratio of 0.06. For features such as σ_{aa} and σ_{af} it can be seen that intersections of modulation types occurred at this ratio or just below this ratio. Thresholds of the decision tree might have been chosen accordingly. This might be one of the reasons for more accurate classification.

When considering all the channel conditions, it can be seen that the maximum decrease in classification accuracy between the biggest and smallest training dataset utilised is 6%. The highest classification error is 20.66%, which is observed when using the smallest training dataset for a fading ratio of 0.4 at an SNR of 10 dB.

4.3 Hardware Implementation

4.3.1 Implementation

For the hardware implementation, very high speed integrated circuit hardware description language (VHDL) was used to describe the behaviour of the design and a synthesis tool was used to map the architecture. The design was implemented in Xilinx's Vivado 2016.2 environment. The feature extraction concept design was translated into VHDL code for implementation where its functionality was confirmed and evaluated in the hardware domain. The feature extraction module was designed and its functionality was tested in simulation. After satisfactory results were obtained, design constraints were added and a synthesis of the design was performed in order to evaluate the size of the design and performance of its functionality. After acceptable results were obtained, the routing of the design was done. If timing requirements were not met, placement and routing as well as the description of the design's behaviour had to be improved. After all the requirements were met, a bitstream was generated and loaded onto the FPGA. The process is illustrated in Figure 96 [123].

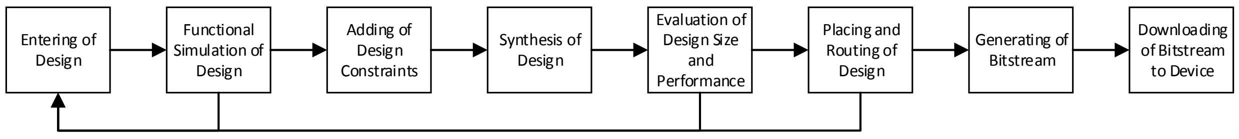

Entering of Design
Functional Simulation of Design
Adding of Design Constraints
Synthesis of Design
Evaluation of Design Size and Performance
Placing and Routing of Design
Generating of Bitstream
Downloading of Bitstream to Device

Figure 96: Flow Diagram of development of Firmware

Data was transferred between Matlab and the FPGA using User Datagram Protocol (UDP) Internet Protocol (IP). Feature extraction was performed on the data received by the FPGA. The feature values were sent back to Matlab and were used for evaluation or classification by the decision tree discussed in 4.2.2.6.

The extraction of the features in hardware was designed according to the specifications of a front-end processor provided by the Council for Scientific and Industrial Research (CSIR)[1]. It was designed with the prospect of future integration with the provided front-end processor. The analysis of some front-end processor components as well as interaction with the front-end processor is first discussed. The transfer of data between Matlab and the FPGA is also explained. The design and implementation of feature extraction is then shown and described.

4.3.1.1 Interaction with front-end processor

The front-end processor provided by the CSIR is similar to the front-end processor described in Chapter 3, with additional features and improved performance. The front-end processor is however designed for high, agile bandwidth signals and operates at a sample frequency of 4GHz. A great amount of memory is used to generate signals for the front-end processor at the full bandwidth, whereafter a filter and decimation significantly reduces the data rate. This method of testing is prohibitive and a better simulation approach was thus required. The effect of the front end processing within the frequency band of interest was therefore analysed instead to determine whether it is negligible. I&Q samples were then generated accordingly in Matlab. These samples, calculated from signals with the same signal parameters mentioned in section 4.1 were then used as input for the feature extraction block within the hardware. The signal parameters were kept the same in order to compare the feature values calculated in hardware with the feature values calculated in software.

4.3.1.1.1 Hilbert Filter

The effect of the Hilbert filter on a signal was analysed in Matlab simulation. The frequency response of the filter was investigated with the aim of deciding whether to neglect the effects of the filter or not.

A linear time-invariant (LTI) system can be fully characterised by its impulse response $h[n]$ in the time domain. Given an input $x[n]$, the output $y[n]$ is given by the convolution sum [115]:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] \tag{53}$$

The frequency response $H(e^{j\omega})$ is directly related to the impulse response through the Fourier transform. The Fourier transform of an output is given by:

$$Y(e^{j\omega}) = H(e^{j\omega})X(e^{j\omega}) \tag{54}$$

Where $Y(e^{j\omega})$ and $X(e^{j\omega})$ are the Fourier transforms of $y[n]$ and $x[n]$ respectively. In the polar form, the Fourier transform of the output is given by:

$$|Y(e^{j\omega})| = |H(e^{j\omega})| \cdot |X(e^{j\omega})| \tag{55}$$

$$\angle Y(e^{j\omega}) = \angle H(e^{j\omega}) + \angle X(e^{j\omega}) \tag{56}$$

[1] The front-end processor was developed by the Radar and Electronic Warfare Competency Area in the Defence, Peace, Safety and Security (DPSS) unit.

Where $|H(e^{j\omega})|$ is the gain or magnitude response of the LTI system and $\angle H(e^{j\omega})$ the phase shift or phase response. The effects of the magnitude and phase on a signal are known as magnitude - and phase distortions. The group delay $\tau(\omega)$ is the derivative of the phase response and is given by:

$$\tau(\omega) = grd[H(e^{j\omega})] = -\frac{d}{d\omega}\{\angle H(e^{j\omega})\} \tag{57}$$

The group delay is used to describe the linearity of the phase. Noting that a delay in time is related to phase that is linear with frequency, the effect of delay can be characterised. A wideband signal can be seen as the superposition of narrowband signals with different centre frequencies. If the group delay of a wideband signal is constant with frequency, the delay for all of the narrowband signals will be identical. If the group delay is however not constant with frequency, the narrowband signals at different frequencies will undergo different delays which will result in time dispersion of the energy of the output signal. Phase nonlinearity thus causes time dispersion.

If the impulse response of an ideal delay system is given by:

$$h[n] = \delta[n - n_d] \tag{58}$$

where n_d is the time delay. The frequency response is given by:

$$H(e^{j\omega}) = e^{-j\omega n_d} \tag{59}$$

such that

$$|H(e^{j\omega})| = 1 \tag{60}$$

$$\angle H(e^{j\omega}) = -\omega n_d \tag{61}$$

Figure 97 shows the frequency responses of an ideal delay system with a time delay of $n_d = 16$.

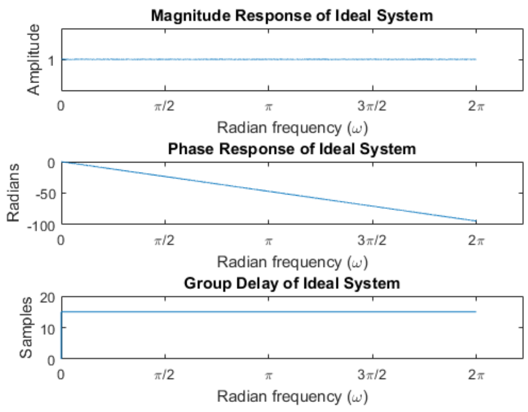


Figure 97: Frequency Response of an Ideal Delay System at n_d = 16

A Parks McClellan Hilbert FIR filter of order M equal to 30 is used for I&Q demodulation in the front-end processor. The coefficients of the filter were generated in Matlab. Because the filter has an even order and odd symmetry of coefficients, the filter is classified as a Type 3 FIR filter.

For a generalised linear-phase system, the frequency response is given by:

$$H(e^{j\omega}) = A(e^{j\omega})e^{-j\alpha\omega+j\beta} \tag{62}$$

such that the group delay is:

$$\tau(\omega) = \alpha \tag{63}$$

and the linear phase is:

$$\angle H(e^{j\omega}) = \beta - \omega\alpha \tag{64}$$

where α and β are constants and $0 < \omega < \pi$. For a Type 3 FIR filter, the frequency response has the form:

$$H(e^{j\omega}) = A_0(e^{j\omega})e^{-j\omega\frac{M}{2}} \tag{65}$$

For the analysis of the Hilbert filter, a Dirac delta function was applied to the filter. Figure 98 shows the frequency response of the Dirac delta function at time delay $n_d = 0$. Figure 99 shows the frequency response of the filter at time $n_d = 0$ and Figure 100 shows the frequency response after the effect of filter delay for $n_d = 0$ had been removed.

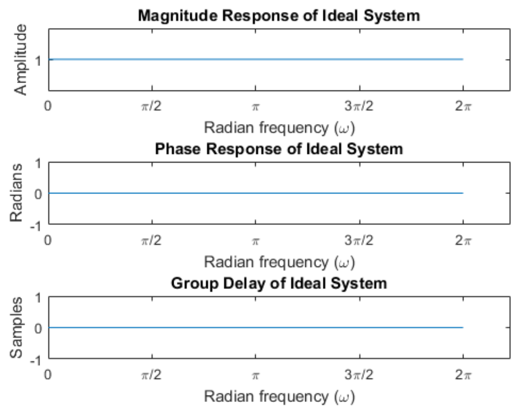


Figure 98: Frequency response of Dirac delta function at $n_d = 0$

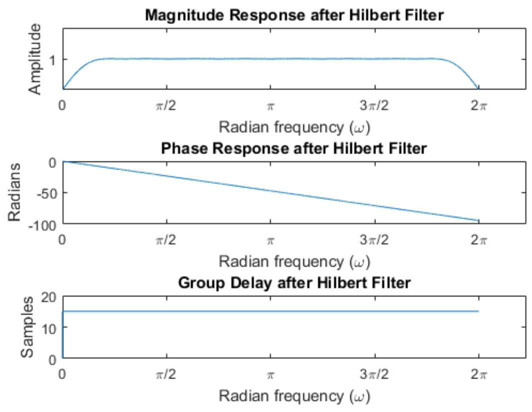


Figure 99: Frequency Response of Dirac Delta Function at $n_d = 0$ with Effect of Filter Delay

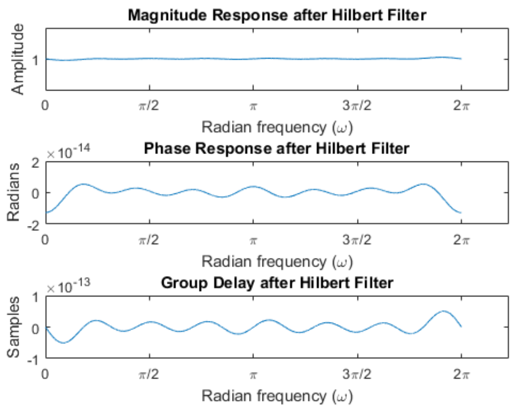


Figure 100: Frequency Response of Dirac Delta Function at $n_d = 0$ with Filter Delay Removed

For $M = 30$, it can be seen that the filter satisfies equation (65). With the delay removed, the magnitude- and phase response effects are negligibly small. By comparing Figure 98 and Figure 99 it can be seen that the filter shows desired results in the passband, $f_{p1} < f < f_{p2}$. The group delay is also constant and the filter has a linear phase with no effect of time dispersion. It can therefore be assumed that all narrowband signals in the given bandwidth will undergo identical delay. It can also be concluded that the magnitude- and phase response effects can be neglected, due to the fact that these effects are negligibly small compared to other signal effects such as noise and quantisation which dominate these errors.

4.3.1.1.2 Quantisation

The operation of quantisation can be represented by [126]:

$$\hat{x}[n] = Q(x[n]) \tag{66}$$

Where $x[n]$ is the input sample and $\hat{x}[n]$ the quantised sample. A $(B+1)$-bit quantiser generally has 2^{B+1} quantised levels. The most significant bit is considered as the sign bit in a two's-complement system. The remaining bits in the code word represent the value. The parameter, X_m, determines the full scale level of an ADC. The step size of the quantised levels depends on X_m and is given by:

$$\Delta = \frac{2X_m}{2^{B+1}} = \frac{X_m}{2^B} \tag{67}$$

The quantisation error is given by:

$$e[n] = \hat{x}[n] - x[n] \tag{68}$$

The quantisation error samples are uniformly distributed random variables and can be seen as additive white-noise. If sample values are rounded by quantisers to the nearest quantisation level, the quantisation noise samples are in the range

$$-\Delta/2 \leq e[n] < \Delta/2 \tag{69}$$

This is only true if

$$(-X_m - \Delta/2) < x[n] \leq (X_m - \Delta/2) \tag{70}$$

If $x[n]$ is outside this range the values are clipped and the errors may be larger.

With $X_m = 1$ and a 10-bit ADC from (67):

$$\Delta = \frac{1}{2^9} = 1.953 \times 10^{-3} \tag{71}$$

and from (69):

$$-9.766 \times 10^{-4} \leq e[n] < 9.766 \times 10^{-4} \tag{72}$$

The signal-to-quantisation noise ratio (SNR_Q) of a $(B+1)$-bit uniform quantiser is given by:

$$SNR_Q = 10log_{10}\left(\frac{{\sigma_x}^2}{{\sigma_e}^2}\right) = 6.02B + 10.8 - 20log_{10}\left(\frac{X_m}{\sigma_x}\right) \tag{73}$$

where σ_x is the RMS value of the amplitude of a signal and ${\sigma_e}^2$ is the noise variance or noise power given by:

$$P_{ee}(e^{j\omega}) = {\sigma_e}^2 = \frac{\Delta^2}{12} = \frac{2^{-2B}{X_m}^2}{12}, \quad |\omega| \leq \pi \tag{74}$$

If a cosine wave with amplitude of 1 is considered with $X_m = 1$, then $\sigma_x = \frac{1}{\sqrt{2}}$ and

$$SNR_Q = 6.02(9) + 10.8 - 20log_{10}\left(\frac{1}{0.7071}\right) \tag{75}$$
$$SNR_Q = 61.97\ dB$$

given that the amplitude doesn't exceed X_m. From equation (72) and (75), it was concluded that the quantisation noise is much smaller than the signal to noise ratio of the investigated signals, and is thus negligible.

4.3.1.1.3 Instantaneous Amplitude, Phase and Frequency

The CSIR implemented front-end processor also supplies the instantaneous amplitude, phase and frequency of the signal within its band. It was therefore only necessary to implement the calculation of the features based on the instantaneous information. The conversion of the I&Q signal samples to polar form of the front-end processor was analysed and the instantaneous amplitude, phase and frequency samples were generated accordingly.

4.3.1.2 *Data Transfer between Matlab and FPGA*

The Matlab function and FPGA process to transfer the data were provided and were not developed. The data thus only had to be converted to the correct form to be transferred. The instantaneous amplitude, phase and frequency samples of a signal were first converted from double-precision floating point values to scaled 16 bit integers. The samples were scaled according to the values that would have been obtained by using the front-end processor. Each instantaneous amplitude, phase or frequency sample consisted of 16 bits and was type casted to two 8-bit unsigned integers as this is the data size required for transfer to the hardware system.

The converted instantaneous amplitude, phase and frequency samples were sent in UDP packets to the FPGA and were written into three different memory blocks respectively. UDP packets contained 16 words where each word consisted of 512 bits, or equivalently, 64 unsigned bytes. A word thus contained 32 samples. After all the data was sent, a final packet, containing one 32 bit word, was sent to the FPGA as a notification that all data had been sent. This notification was used to initiate the calculations of the features. After the notification was received, the samples were read from memory. In normal operation of the front-end processor, the data is parallelised by a factor of 32 between the ADC and the FPGA, so each clock cycle contains 32 signal samples when processed. Thirty two samples were thus read each clock cycle and used for the calculations. Once the calculations were done, the feature values were stored in a register and sent back to Matlab. Figure 101 shows a flow diagram of the transfer of data.

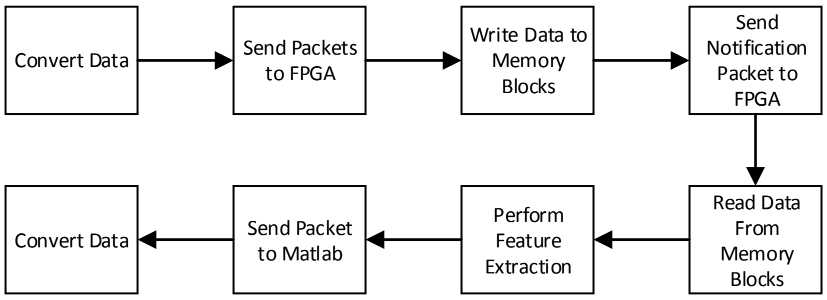

Convert Data
Send Packets to FPGA
Write Data to Memory Blocks
Send Notification Packet to FPGA
Read Data From Memory Blocks
Perform Feature Extraction
Send Packet to Matlab
Convert Data

Figure 101: Flow Diagram of Data Transfer between Matlab and FPGA

4.3.1.3 Behavioural Design Considerations

A Virtex7 VX690TFFG1930-2 FPGA was used for the implementation. DSP slices in an FPGA can be used for multiple parallel math operations such as multiplications, accumulate, add, etc. The Virtex7 VX690TFFG1930-2 FPGA contains 3600 DSP48E1 Slices. A block diagram of a DSP48E1 Slice can be seen in Figure 102. The math portion of the DSP slice includes a 25-bit by 18-bit, two's compliment multiplier and a 48-bit accumulator. The result of the multiplier is a 43-bit output that is sign-extended to 48-bits. An adder/subtracter can have three 48-bit inputs and results in a 48-bit output [125]. If the inputs are greater than the specified widths, one or more DSP slices are cascaded for the math operation. It is thus desirable to keep the inputs within the specified widths to avoid the unnecessary overutilisation of limited device resources.

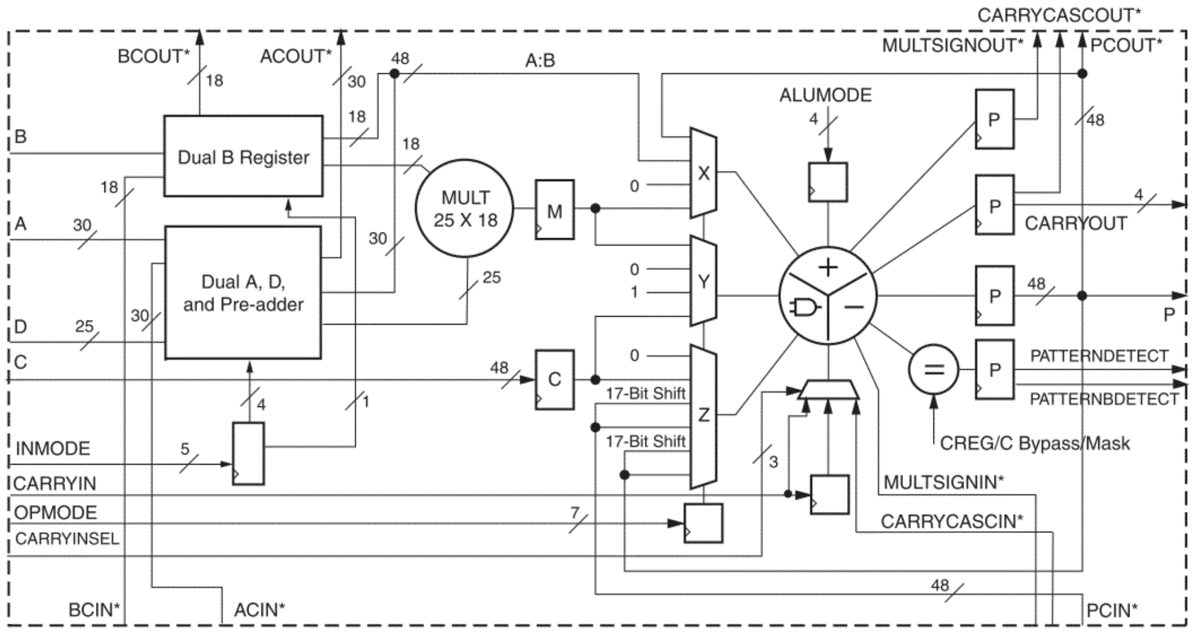


Figure 102: DSP48E1 Slice [125]

For a DSP slice to operate at full speed (600 MHz), pipeline stages must be implemented by means of registers. For a multiply operation, three-stage pipelining is suggested. For non-multiply operations, two-stage pipelining is suggested. There is thus a trade-off between increased clock frequency and data latency [125].

4.3.1.4 Calculation of the features

The feature calculations were performed in cascade to improve calculation speed. The feature extraction entity was designed to use thirty two cascaded paths, since 32 samples will be received each clock cycle from the front-end processor. Shorter signals consume less hardware resources, and reduce calculation time. Since transmitters in a non-cooperative environment can change their modulation type quickly, it is desirable to classify modulation types using the fewest samples possible for quicker classification turnaround. The calculation of the features is also a subsystem that will from part of a greater system. The utilisation of device resources should thus also be kept to a minimum. From the results obtained in section 4.2.2.5, it can be seen that most feature values remain separable for the number of samples greater than 1920. A number of samples greater than 1920 will thus be adequate. 2048 samples were chosen, since it is a power of 2 and simplifies many calculations.

If the feature values are calculated over 2048 samples, at least 64 clock cycles are needed by each math operation to process all 2048 samples. Independent operations, such as the calculations of the features based on the instantaneous amplitude, phase and frequency respectively, were performed in parallel. The dependent operations, such as the calculations to obtain the individual features, were performed sequentially. Many of the features consisted of intermediary calculation steps that were identical and calculated values could thus be re-used. Pipelining stages were implemented to ensure that the timing constraints were met. Some general operations include the multiplication, division, the calculation of the average, summation, the square root and squared power. These operations are explained in detail below:

- For multiplication
    - First the bit width of a value was checked to determine whether it exceeded the bit width of a multiplier. The bit width was determined by finding the position of the most significant bit.
    - If the bit width of the values exceeded the bit width of the input to a multiplier, the value was shifted right by the number of positions exceeding the bit width of the input. This was equivalent to dividing the value by a power of two where the power is equal to the number of positions it was shifted. This was done in order to reduce the number of multipliers used for a single multiplication operation.
    - After the multiplication was performed, the result is shifted left by the number of positions the two input values were shifted right.
    - It can be noted that there will be a decrease in accuracy. There is thus a trade-off between device resources and precision.
- For division, the denominator was inverted and multiplication was performed.
- The summation of thirty two values was performed by using an adder tree:
    - The depth of the tree is equal to the $\log_2$ of the number of inputs.
    - The tree starts with the number of branches equal to the number of inputs and ends with one branch.
    - Values are added together in pairs of two each clock cycle until only one value remains.
    - The number of branches is thus halved each clock cycle.

- The average was calculated as follows:
    - The average of the 32 values received in each clock cycle is calculated by adding the values together and dividing it by 32. The calculation is referred to as Parallel Mean in Figure 103 to Figure 106. An adder tree was used to add the values. The division was executed by right shifting the sum 5 positions right which is equivalent to division of 32.
    - The calculations in the latter step were performed 64 times which resulted in 64 averages over 64 clock cycles. The average of the 64 values was calculated by accumulating the values each clock cycle and dividing the answer by 64 when 64 clock cycles had passed. For this second average operation, right shifting was not used. If the number of samples utilised for the calculation of the features would change and would not be divisible by a power of two, the operation would not work. A counter was used instead to determine the value by which the accumulated value should be divided by. The calculation is referred to as Serial Mean in Figure 103 to Figure 106.
- An IP component was used to calculate the square root.

The adder-tree, average and divider were already developed generic entities and were not developed for the implementation of the feature extraction. The calculations of the normalised-centred amplitude, phase and frequency are shown in Figure 103. The values were shifted left (or multiplied) whenever division was performed. This was done in order to retain precision of the values. It can be noted that the frequency samples were not normalised by the sampling frequency to prevent loss of precision.

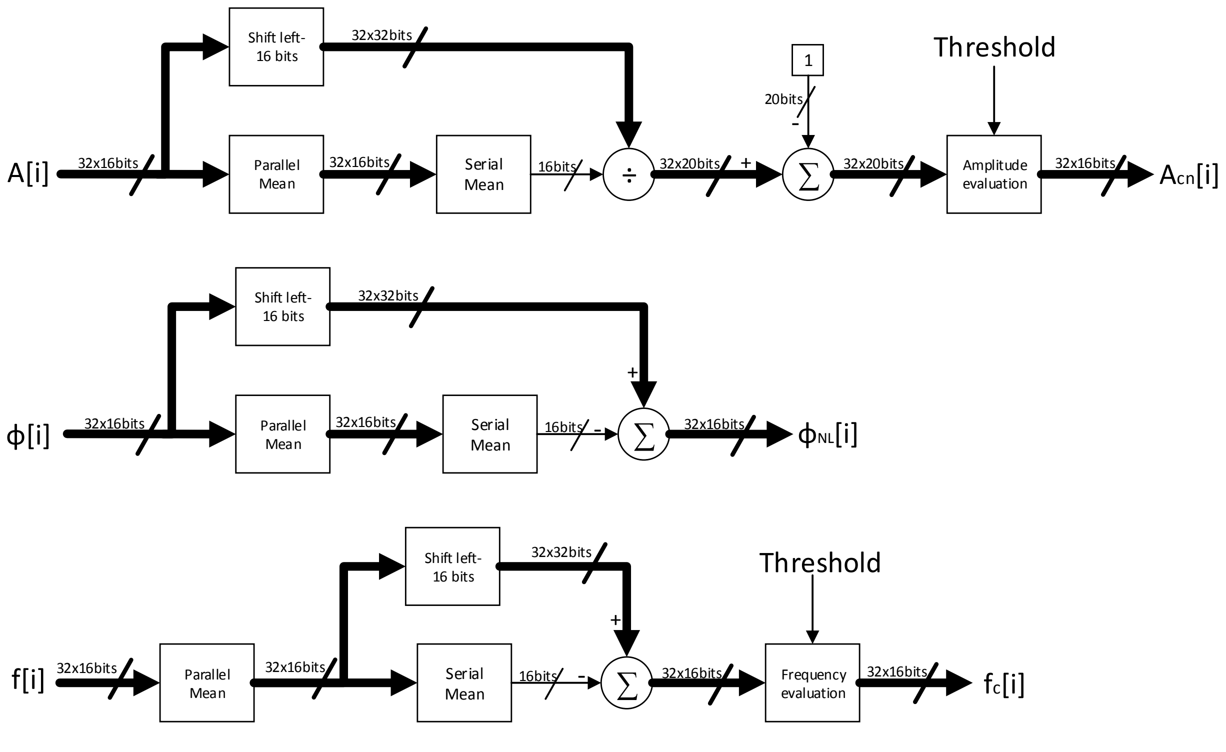


Figure 103: Flow Diagram for normalising and centring the Instantaneous Amplitude, Phase and Frequency in Firmware

For the features based on the instantaneous amplitude, phase and frequency, the variance needed to be calculated for various features. Generic blocks were thus created to calculate averages of squared values as well as the squared values of averages. These blocks were then implemented where required for the different features.

For the calculation of the μ_{42}^{a} and μ_{42}^{f}, which will be described in the following diagrams, the squared values resulted in 32 bits. When these values have to be squared again, they exceed the bit width of a multiplier. The method discussed for multiplication was applied here. Since a value is multiplied by itself, the maximum bit width of the value cannot exceed the maximum bit width of the smallest input to a multiplier, which is 18 bits. The values were thus scaled to 18 bits, multiplied and scaled back to their original size. After the fourth powers of the values were calculated, their average was calculated. As mentioned earlier, the calculation of the average requires the summation of the thirty two values. Since values of different sizes were added, each value could not be scaled individually according to the input width of an adder. The bits of the values were however divided into two parts in order to calculate the average of the 64 averages since a multiplier was required to calculate the average. The calculation of the average was designed to restrict the input in order to ensure the utilisation of a single multiplier for an operation. The average of the most significant bits and least significant bits were calculated separately. The answer of the most significant bits were then shifted and added to the answer of the least significant bits. It can again be noted that there is a loss in accuracy and also a trade-off between device resources and precision.

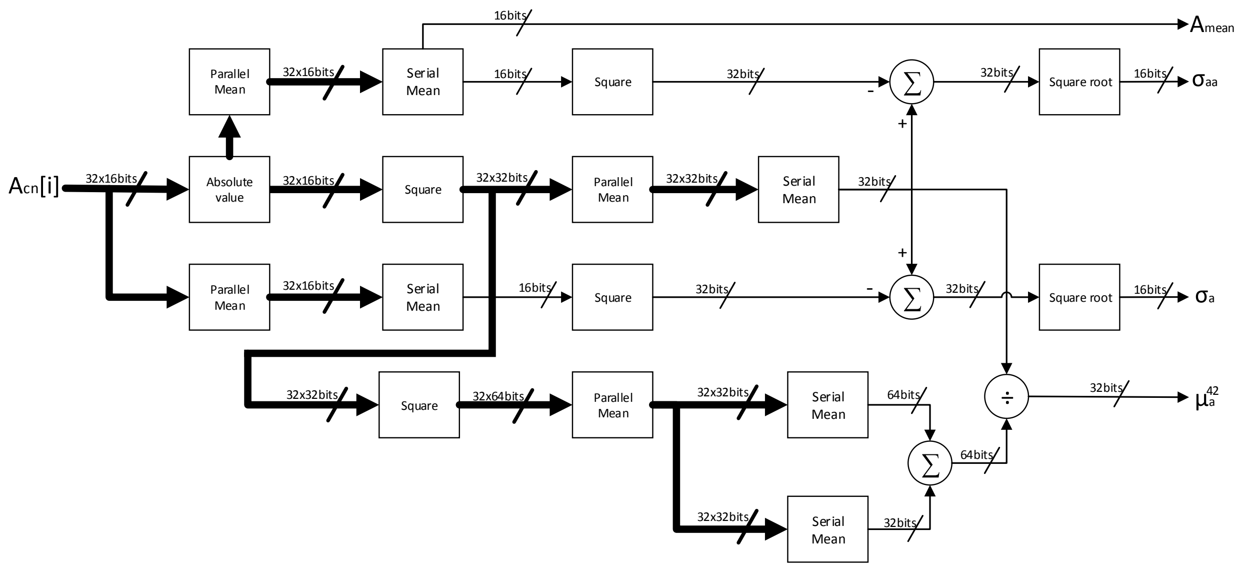


Figure 104: Flow Diagram for Instantaneous Amplitude Based Feature Extraction in Firmware

The flow diagram in Figure 104 illustrates the following calculations to extract features A_{mean}, σ_{aa}, σ_a and μ_{42}^a:

- For each clock cycle the absolute values of 32 normalised-centred instantaneous amplitude samples were first obtained.
- The average of the absolute values received in each clock cycle was calculated and resulted in A_{mean}.
- The squared values of the average of both direct and absolute amplitude were calculated.
- The average of the squared values of the amplitude was calculated. Either the direct value or the absolute value could be used for calculations since the square of the values were used.
- The averages of squared values from both the absolute and direct values were subtracted from the squared values averaged individually. The two answers resulted in the variance of the direct and absolute values respectively.
- The square root of each was calculated to give σ_{aa} and σ_a.
- The fourth power of either the direct or absolute value calculated by computing the squared value twice.
- The average of the fourth power values was calculated.
- The average of the fourth power values were divided by the squared value of the average of the squared values to give μ_{42}^a.

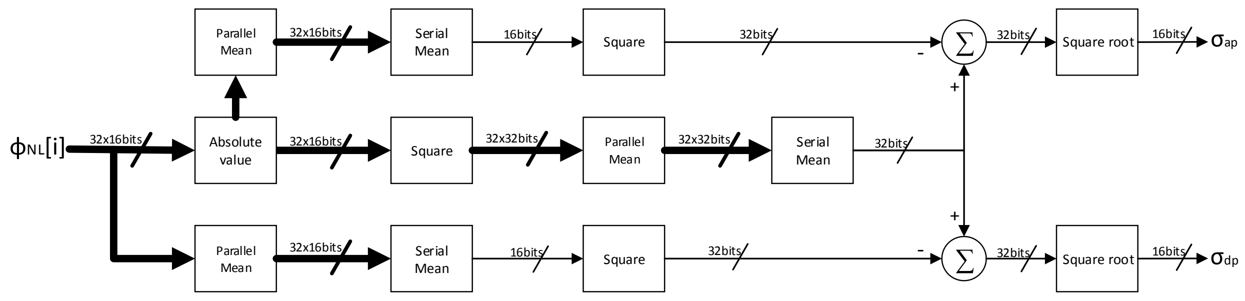


Figure 105: Flow Diagram for Instantaneous Phase Based Feature Extraction in Firmware

The flow diagram in Figure 105 explains the following calculations to extract features σ_{ap} and σ_{dp}:

- For each clock cycle the absolute values of 32 non-linear centred instantaneous phase samples were first obtained.
- The squared values of the average of both direct and absolute phase were calculated.
- The average of the squared values of the phase was calculated. Either the direct value or the absolute value could be used for calculations since the square of the values were used.
- The averages of squared values from both the absolute and direct values were subtracted from the squared values averaged individually. The two answers resulted in the variance of the direct and absolute values respectively.
- The square root of each was calculated to give σ_{ap} and σ_{dp}.

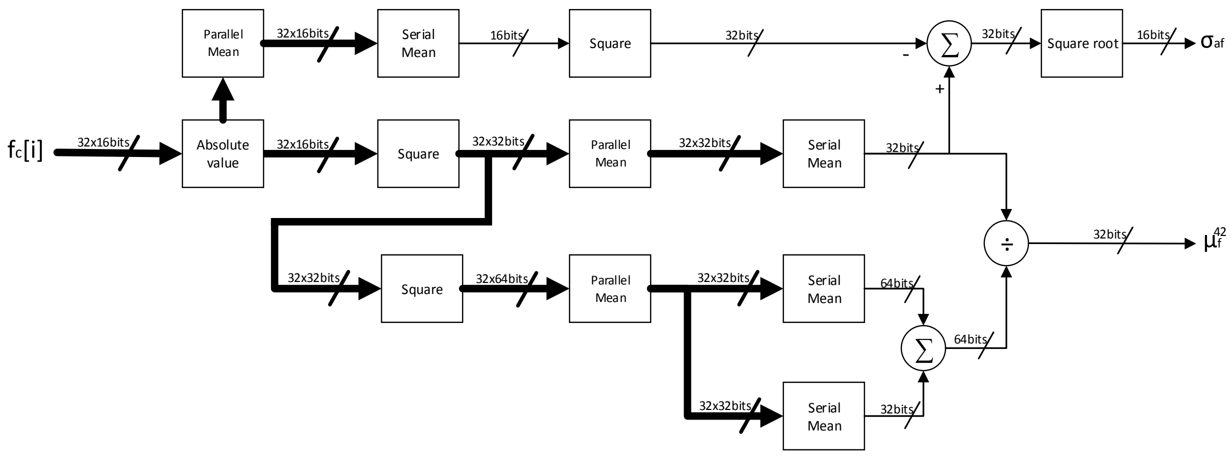


Figure 106: Flow Diagram for Instantaneous Frequency Based Feature Extraction in Firmware

The flow diagram in Figure 106 explains the following calculations to extract features σ_{af} and μ_{42}^{f}:

- In each clock cycle the absolute values of 32 normalised-centred instantaneous frequency samples were first obtained.
- The squared values of the average of both direct and absolute frequency were calculated.
- The average of the squared values of the frequency was calculated. Either the direct value or the absolute value could be used for calculations since the square of the values were used.
- The averages of squared values from the absolute were subtracted from the squared values averaged individually. The answer resulted in the variance of the absolute values.
- The square root was calculated to give σ_{af}.
- The fourth power of either the direct or absolute value calculated by computing the squared value twice.
- The average of the fourth power values was calculated.
- The average of the fourth power values were divided by the squared value of the average of the squared values to give μ_{42}^{f}.

It can be seen that many of the operations have redundant bits and that there is room for improvement of the design. Full functionality of the hardware implementation was however demonstrated.

4.3.2 Results

4.3.2.1 Comparison between simulation and hardware results

The instantaneous information of the signals in section 4.2.2.4 was used for testing in which 2048 samples were used for feature calculation. Figure 107 to Figure 122 show the results of the feature values under flat fading conditions for both 10 dB and 30 dB SNR. The results from simulation (using 144 000 samples) and hardware implementation (using 2048 samples) are presented next to each other for easier one to one comparison in Appendix A.2.

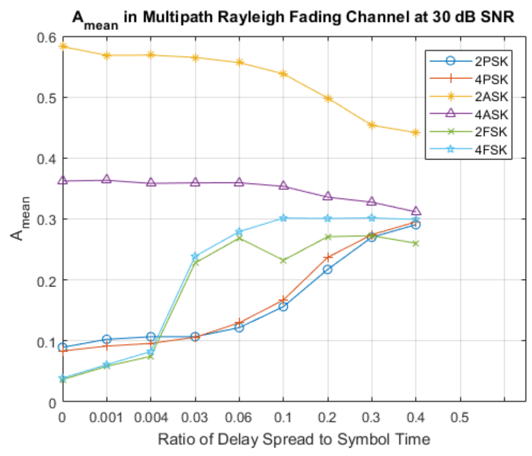


Figure 107: Feature values of A_{mean} in a Flat Fading Channel at 30 dB SNR for Hardware Implementation

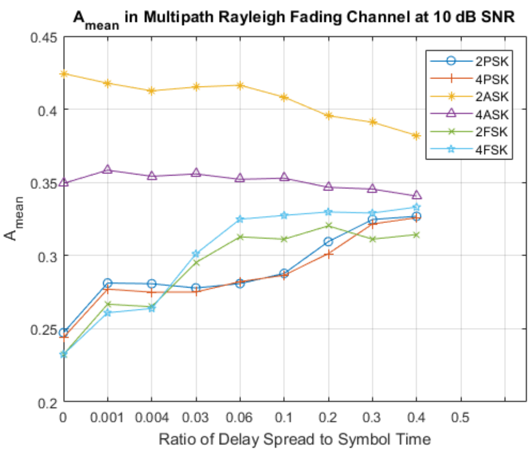


Figure 108: Feature values of A_{mean} in a Flat Fading Channel at 10 dB SNR for Hardware Implementation

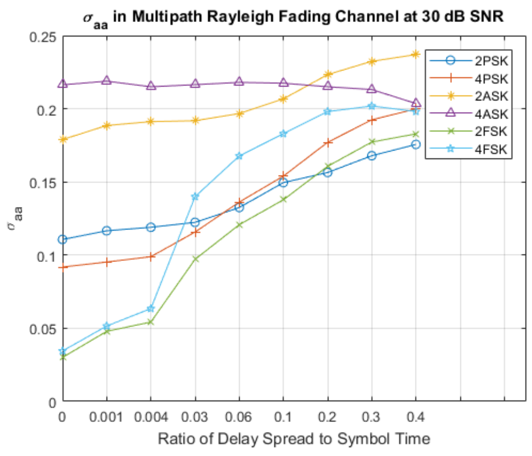


Figure 109: Feature values of σ_{aa} in a Flat Fading Channel at 30 dB SNR for Hardware Implementation

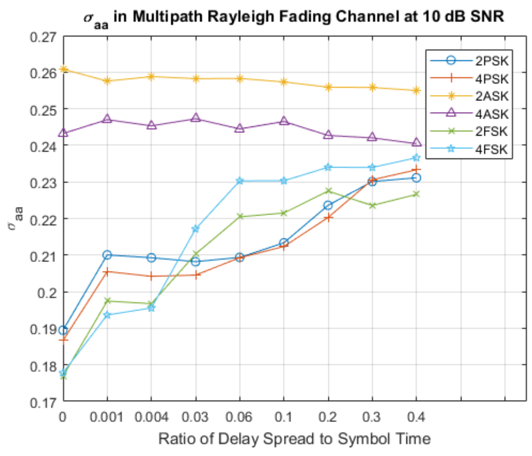


Figure 110: Feature values of σ_{aa} in a Flat Fading Channel at 10 dB SNR for Hardware Implementation

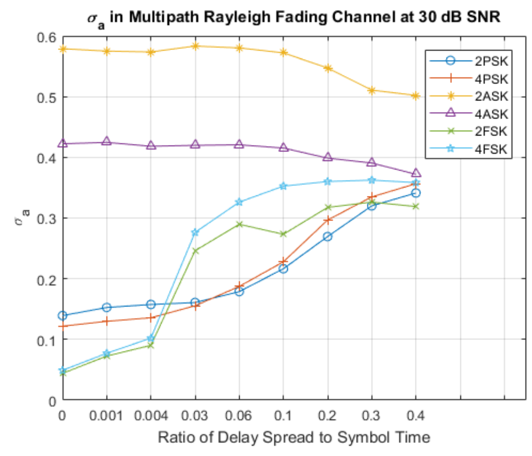


Figure 111: Feature values of σ_a in a Flat Fading Channel at 30 dB SNR for Hardware Implementation

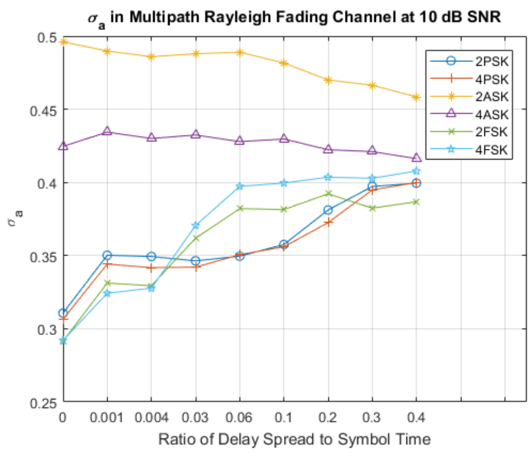


Figure 112: Feature values of σ_a in a Flat Fading Channel at 10 dB SNR for Hardware Implementation

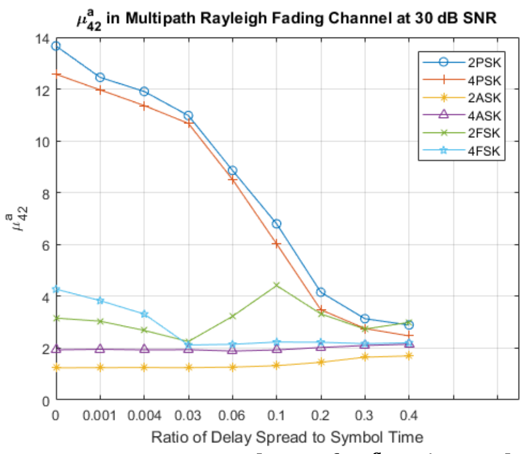


Figure 113: Feature values of μ^a_{42} in a Flat Fading Channel at 30 dB SNR for Hardware Implementation

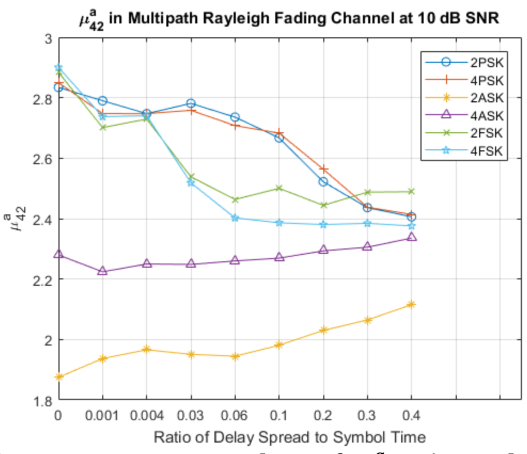


Figure 114: Feature values of μ^a_{42} in a Flat Fading Channel at 10 dB SNR for Hardware Implementation

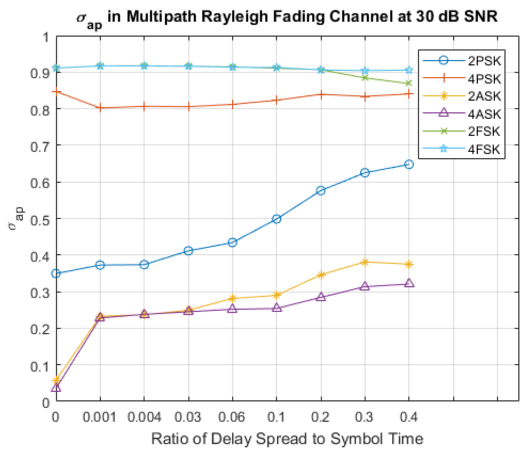


Figure 115: Feature values of σ_{ap} in a Flat Fading Channel at 30 dB SNR for Hardware Implementation

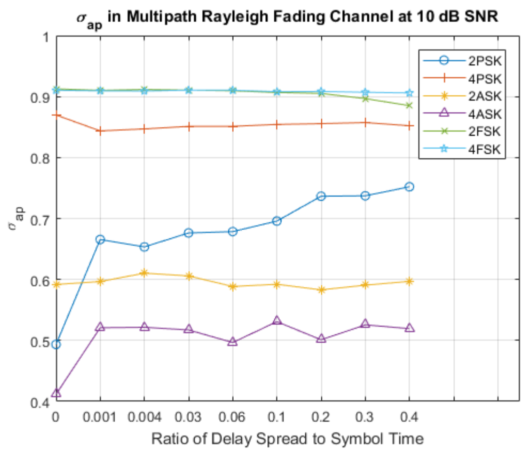


Figure 116: Feature values of σ_{ap} in a Flat Fading Channel at 10 dB SNR for Hardware Implementation

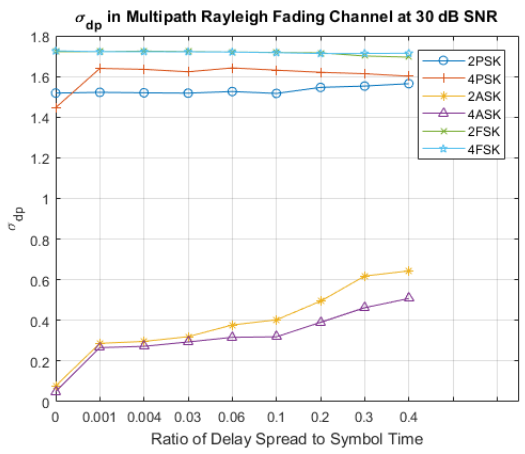


Figure 117: Feature values of σ_{dp} in a Flat Fading Channel at 30 dB SNR for Hardware Implementation

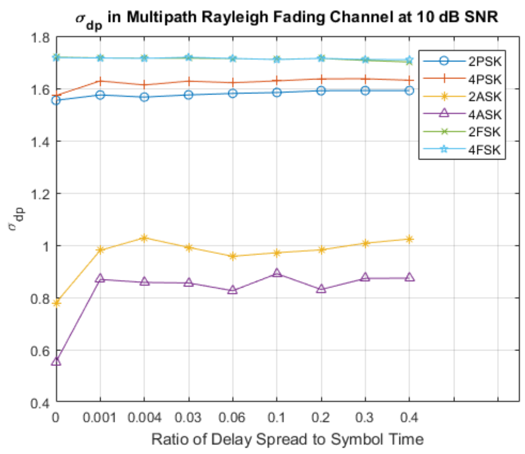
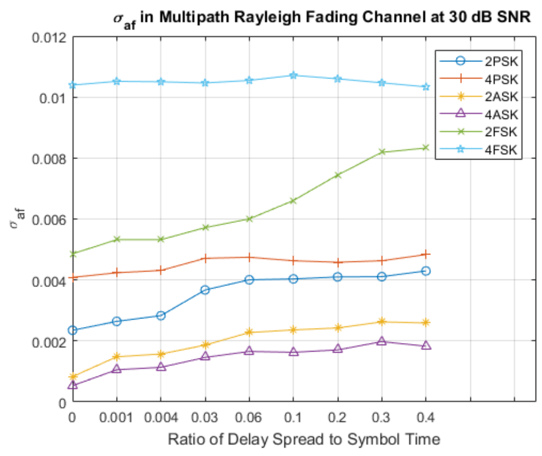


Figure 118: Feature values of σ_{dp} in a Flat Fading Channel at 10 dB SNR for Hardware Implementation



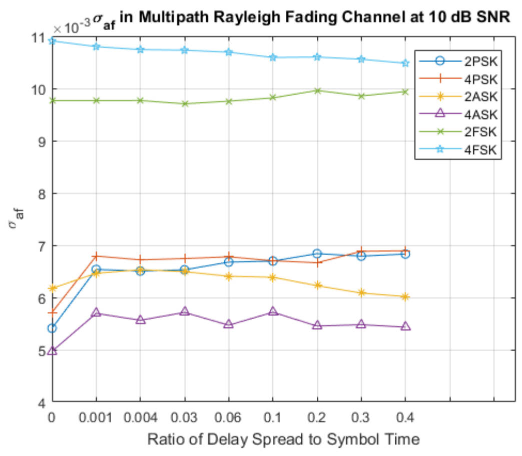
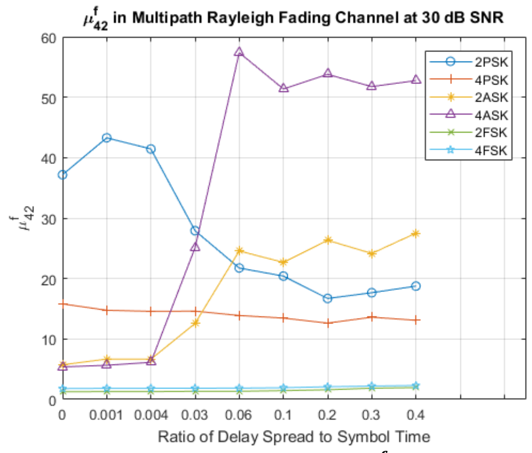


Figure 119: Feature values of σ_{af} in a Flat Fading Channel at 30 dB SNR for Hardware Implementation

Figure 120: Feature values of σ_{af} in a Flat Fading Channel at 10 dB SNR for Hardware Implementation



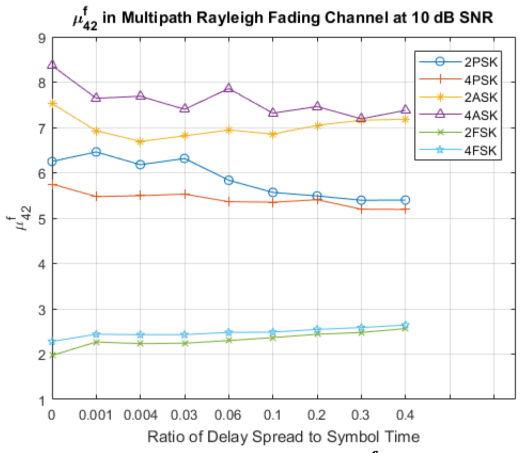


Figure 121: Feature values of μ_{42}^{f} in a Flat Fading Channel at 30 dB SNR for Hardware Implementation

Figure 122: Feature values of μ_{42}^{f} in a Flat Fading Channel at 10 dB SNR for Hardware Implementation

The results show that:

For A_{mean}:

- At 30 dB 2ASK had slightly smaller values than the values obtained in the software results. It can be seen in Figure 69 that the values of 2ASK decreased as the signal length decreased. The values of the other modulation types were very similar to the values obtained in software.
- At 10 dB the values of 2ASK were also slightly smaller than the values obtained in the software results. 4ASK also showed small variations from values in the software results. Figure 70 shows the decrease in the values of MASK with decrease in signal length. It can be seen that 2ASK decreased more rapidly than 4ASK and therefore the values of 2ASK vary more than 4ASK in the hardware results from the software results. The values of MPSK and MFSK were very similar to the values obtained in software.

For σ_{aa}:

- At 30 dB SNR MASK had larger values in the hardware results than the values calculated in software. The values of 4ASK differ only by a small amount while 2ASK had increased significantly. In Figure 71 it can be seen that the minimum values of 2ASK increased as the number of samples decreased. The variance of the hardware and software results did however decrease for fading ratios above 0.2. The values of 2PSK also vary from the values obtained in software at fading ratios above 0.1. The values of 2FSK also show variations at fading ratios from 0.03 to 0.2.
- At 10 dB SNR the values of all the modulation types were slightly higher than the values of the software results. Figure 72 shows however that the decrease in signal length was not the reason for the decrease in the feature values.

For σ_{a}

- At 30 dB SNR the values of 2ASK was smaller than the values obtained in software. The other modulation types had very similar results than the results obtained in software. From Figure 73 it can be seen than the values of 2ASK decreased as the signal length decreased.
- At 10 dB SNR the values of 2ASK were again smaller than the values obtained in software. The values of 4ASK were also slightly smaller. In Figure 74 it can be seen that the values of 2ASK decreased notably as the signal length decreased and 4ASK also decreased slightly as the signal length decreased. The values of MPSK and MFSK were almost identical to the values obtained in software.

For μ_{42}^{a}

- At 30 dB SNR the values of MPSK and MFSK were notably smaller at lower fading ratios, while the values of MFSK were almost identical to the values obtained in software. The decrease in the values of 2PSK is unexpected given that its values increased as signal length decreased as observed in Figure 75. The trade-off between calculation accuracy and device resources can thus be seen here.
- At 10 dB SNR the values of MFSK and MASK were also higher than the values obtained in software. In Figure 76 it can be seen that the values of MASK increased as the signal length decreased.

For σ_{ap}

- At 30 dB SNR the value of 4PSK is slightly smaller than the values obtained in software. The values of 2PSK and MASK were however significantly larger than the software results for the lower fading ratios. The variance in the values decreased at larger fading ratios. In Figure 77 it can be seen that the values of 4PSK decreased as the signal length decreased and the minimum values of 2PSK increased as the signal length decreased. It is however interesting to see that the values of MASK remained almost constant as the signal length decreased, but differences were observed between hardware and software results. Although the modulation types remain separable the increase and decrease in values might lead to misclassification when values obtained from hardware are used.

- At 10 dB SNR the values of both 4PSK and 2PSK were smaller than the software results while the values of MFSK and MASK almost remained constant. The values of 2ASK were however smaller than the results obtained in software. In Figure 78 it can be seen that the values of 4PSK decreased as the signal length decreased while the minimum values of 2PSK increased as the signal length decreased. The values of 2ASK also decreased as the as the signal length decreased.

For σ_{dp}

- At 30 dB SNR the values of MPSK and MFSK were smaller than the values obtained in software, while the values of MASK were slightly larger than the software results. In Figure 79 it can be seen that the values of MPSK decreased as the signal length decreased, while and the maximum values of MASK increased slightly as the signal length decreased. It is however interesting to see that the values of MFSK were smaller for the hardware results, while the values remained almost constant as the signal length decreased.
- At 10 dB SNR the values of MSPK were smaller while MASK and MFSK were very similar to the results obtained in software. In Figure 80 it can be seen that the values of MSPK decreased significantly as the signal length decreased.

For σ_{af}

- At 30 dB SNR the values of 2FSK were significantly smaller than the values obtained in software. The other modulation types had almost identical results than the software results. In Figure 81 it can be seen that the minimum values of 2FSK increased significantly as the signal length decreased. Although the values of 4FSK decreased as the signal length decreased, the value remained almost constant for the number of samples equal to or higher than 2048 samples.
- At 10 dB SNR the values of all the modulation types were slightly smaller, except for the values of 4FSK which were notably smaller. In Figure 82 it can be seen that the values of 4FSK decreased as the signal length decreased.

For μ_{42}^{f}

- At 30 dB SNR it can be seen that MASK have very different results from the results obtained in software, while the values of MPSK and MFSK were very similar. In Figure 83 it can be seen that the values of 4ASK decreased significantly for the first decrease in signal length. The great difference in the values of MASK however requires further investigation.
- At 10 dB SNR the values of all the modulation type were very similar to the results obtained in software, except for MASK from a fading ratio of 0.3. In Figure 84 it can be seen that the values of MASK varied with the decrease in signal length.

In summary, the results show that the values correspond to the values obtained in 4.2.2.4 and are very similar for most features. The calculation of the feature values in hardware showed to be feasible. By comparing the figures of the signal length analysis with that of the results obtained in this section, it can be seen that most differences were due to the number of samples used for calculation. Very similar results were obtained for modulation types that were not affected by the number of signals used for calculation of the feature values. For the calculation of μ_{42}^{a} and μ_{42}^{f} much bigger differences were observed. It can thus be seen that measures taken to improve device resource utilisation and computational complexity influenced feature values. The differences in the values of MASK for μ_{42}^{f} required further investigation. For direct comparison the simulations were repeated at the same signal length as that used for hardware, namely 2048 samples. These feature values can be seen in Appendix A.3. Similar results were obtained than for the hardware results for μ_{42}^{f}, although the values of MASK from hardware were slightly smaller. It is thus confirmed that the major factor contributing to the big difference in feature values were due to the number of samples used.

4.3.2.2 *Comparison between simulation and firmware results of decision tree*

From the observations in the results of the previous section, it was not expected that the decision trees constructed in section 4.2.2.6 would work as well for classification when 2048 samples are used to calculate the feature values. The performance of these decision trees was however investigated by using the feature values calculated in hardware. The same set of signals investigated in software simulation for the testing of the decision tree was used for the hardware investigation. Table 15 and Table 16 show the classification accuracy achieved by the decision tree using the hardware results.

Table 15: Classification accuracy (%) of decision trees with decreasing training sets over varying R_{DS} at 30 dB SNR using Feature Values obtained in Hardware

Training vectors		R_{DS}								
		0	**0.001**	**0.004**	**0.03**	**0.06**	**0.1**	**0.2**	**0.3**	**0.4**
Test1	400	71.5	73.6	75.5	76.5	80.17	80.00	75.5	68.17	68.33
Test2	300	71.67	74.17	75.83	75.67	79.83	80.5	76.00	68.83	68.83
Test3	200	72.33	77.33	77.67	76.83	79.17	79.00	76.17	67.33	68.33
Test4	100	70.5	74.00	75.00	75.67	78.50	78.33	74.50	69.00	65.00
Test5	50	70.17	76.00	76.00	77.33	81.17	78.5	76.50	71.00	66.83

Table 16: Classification accuracy (%) of decision trees with decreasing training sets over varying R_{DS} at 10 dB SNR using Feature Values obtained in Hardware

Training vectors		R_{DS}								
		0	**0.001**	**0.004**	**0.03**	**0.06**	**0.1**	**0.2**	**0.3**	**0.4**
Test1	400	71.17	64.33	65.17	64.5	64.17	64.33	61.50	59.50	60.50
Test2	300	72.17	64.33	65.83	63.83	64.00	63.50	61.83	61.33	62.17
Test3	200	71.67	66.17	66.83	65.00	64.17	65.33	62.67	61.83	60.17
Test4	100	75.33	67.33	65.67	64.67	64.50	63.33	64.83	61.83	60.50
Test5	50	73.67	66.00	64.50	64.33	64.83	64.17	64.17	61.17	61.5

From Table 15 and Table 16 it can be seen that that classification accuracy decreased on average by 22.89% and 29.24% for the biggest dataset at an SNR of 30 dB and 10 dB respectively. The classification accuracies are however still much better than random chance (100/6). For 30 dB and 10 dB SNR, the highest and the lowest classification accuracy were 80.5% and 65.00%, and 75.33 and 59.5% respectively. It is however interesting to see that there was not a significant decrease in classification errors when smaller datasets were used. It can also be seen that the classification accuracy increased as the fading conditions deteriorated until a fading ratio of 0.2 for 30 dB SNR. The reason for this behaviour is still unknown and is subject to further investigation.

From the confusion matrices in Appendix B.2, it can be seen that most misclassifications occurred between different orders within modulations of the same type. More specifically, the higher order modulation types were typically misclassified as the lower order modulation types. It can be seen that 4PSK was the most effected by the number of samples utilised for the calculation of the features. For poor fading conditions, 4PSK was almost completely misclassified. The misclassification of 4PSK became worse for 10 dB SNR. The results showed that 4PSK put a limitation on the number of samples required for accurate classification. Future work may include finding features that are more distinguishable on 4PSK.

The decision tree thus had limited performance when a test dataset obtained in hardware with a different signal length was used. It was thus required to construct a decision tree using feature values calculated from 2048 samples for direct comparison with the hardware. Another tree was therefore constructed in Matlab using feature values calculated from 2048 samples, matching the signal lengths used for hardware instead of 144000 samples used in the original tree construction. Because the datasets obtained in hardware did not have full representation of all SNR conditions, only 10 dB and 30 dB SNR, it was necessary to construct a new tree in Matlab with full representation of all the channel conditions. The goal was to determine whether the hardware implementation was feasible for calculating feature values from signals under combined SNR and flat fading conditions. A dataset containing feature values calculated from signals only under SNR conditions was therefore not generated in hardware. Table 17 shows the results of the construction of the trees using sizes of training datasets as discussed in section 4.2.2.6.

Table 17: Results of the construction of the decision trees using 2048 samples

		Before Pruning			After Pruning			
Test	Training vectors	Tree depth (levels)	Re-substitution Error (%)	Cross Validation Error (%)	Tree depth (levels)	Re-substitution Error (%)	Cross Validation Error (%)	Estimated Classification Error (%)
1	60 000	154	3.56	14.00	63	10.25	14.14	12.40
2	45 000	126	3.81	14.29	63	10.38	14.39	12.78
3	30 000	110	3.77	14.22	38	11.59	14.68	13.04
4	15 000	73	4.10	15.07	35	10.93	15.21	14.00
5	7500	47	4.15	16.27	26	10.25	16.01	15.47

It can be seen that the estimated classification errors increased notably compared to the estimated classification errors of the tree constructed in section 4.2.2.6. There is also an increased in the depth, thus the complexity, of the trees.

The tables below show the results of the decision tree for varying flat fading conditions at 30 dB and 10 dB SNR respectively. The top value in each cell is the classification accuracy achieved for feature values obtained from software, indicated with S, and the bottom values in each cell is the classification accuracy achieved for feature values obtained from hardware, indicated with H.

Table 18: Classification accuracy (%) of decision trees with decreasing training sets over varying R_{DS} at 30 dB SNR using Feature Values obtained from 2048 samples

Training vectors	S/H	R_{DS}								
		0	**0.001**	**0.004**	**0.03**	**0.06**	**0.1**	**0.2**	**0.3**	**0.4**
400	S	97.83	97.00	96.50	95.67	96.00	93.00	87.33	80.83	74.17
	H	96.00	94.33	93.17	93.00	93.83	92.17	86.50	80.67	74.50
300	S	97.67	96.83	96.00	94.67	96.00	92.67	87.83	79.00	74.83
	H	98.83	93.83	93.5	92.67	93.67	91.5	85.67	79.33	73.83
200	S	96.17	96.67	95.33	93.5	94.67	91.83	87.17	78.17	74.67
	H	94.83	93.83	94.00	91.00	91.67	90.5	85.17	77.33	74.83
100	S	95.33	95.33	95.7	94.00	94.33	92.67	87.50	77.00	74.83
	H	95.33	95.00	94.67	91.00	91.50	89.50	85.17	76.83	74.17
50	S	93.50	95.00	92.67	91.17	92.83	89.83	83.67	75.33	74.67
	H	93.00	94.50	92.00	91.17	91.67	87.83	83.33	73.83	73.50

Table 19: Classification accuracy (%) of decision trees with decreasing training sets over varying R_{DS} at 10 dB SNR using Feature Values obtained from 2048 samples

Training vectors	S/H	R_{DS}								
		0	**0.001**	**0.004**	**0.03**	**0.06**	**0.1**	**0.2**	**0.3**	**0.4**
400	S	92.33	77.00	79.83	78.00	78.83	73.83	76.17	72.00	69.17
	H	84.50	72.00	72.4	70.83	73.17	71.83	69.33	68.33	65.17
300	S	92.5	77.50	79.33	79.50	79.17	73.5	75.00	73.50	67.33
	H	83.00	71.33	72.32	69.32	74.50	70.50	68.33	68.17	63.67
200	S	91.33	77.17	78.33	77.50	78.67	73.50	73.67	72.00	67.5
	H	84.00	70.67	72.33	70.67	75.50	70.33	69.67	67.33	62.50
100	S	90.50	75.17	77.00	76.50	78.33	73.17	70.00	73.33	65.50
	H	86.67	73.00	75.17	74.33	75.67	70.33	68.17	69.83	64.50
50	S	90.00	73.50	76.67	76.83	76.00	70.00	71.50	69.00	61.83
	H	89.33	72.00	72.17	76.33	75.17	71.33	69.33	66.00	57.17

For 30 dB SNR the results from software and hardware were very similar. At 10 dB SNR the classification accuracies for the hardware results however decreased notably for lower fading ratios. At higher fading ratios the classification accuracies from hardware were closer to the classification accuracies from the software results. Classification accuracy higher than 73% and 57% was achieved for all training datasets for 30 dB and 10 dB SNR respectively. For a full training dataset, classification accuracy higher than 74% and 65% was achieved for 30 dB and 10 dB SNR respectively. Classification accuracy higher than 83% was achieved for all training datasets up to a fading ratio of 0.2 for 30 dB

SNR. For 10 dB SNR, classification accuracy higher than 70% was achieved for al training datasets up to a fading ratio of 0.1. It can be seen that the classification accuracy decreased notably for fading ratios above 0.3.

The following figures and Appendix B.3 show the results of the decision tree of the hardware results for varying flat fading conditions at 30 dB and 10 dB SNR respectively. The decision tree was constructed with a full training dataset.

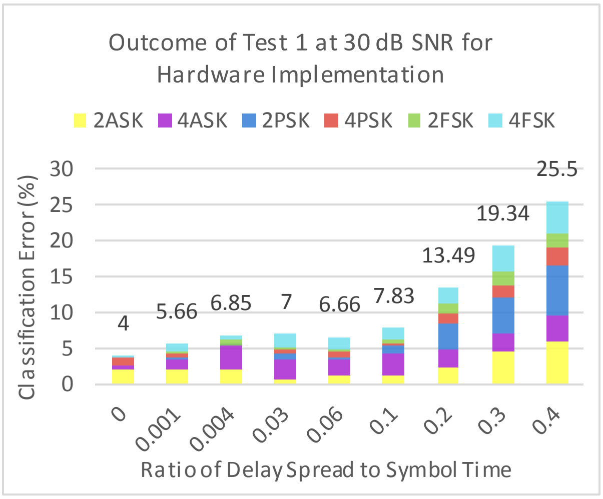


Figure 123: Classification Error for 400 training vectors under varying flat fading conditions at 30 dB SNR for Hardware Implementation

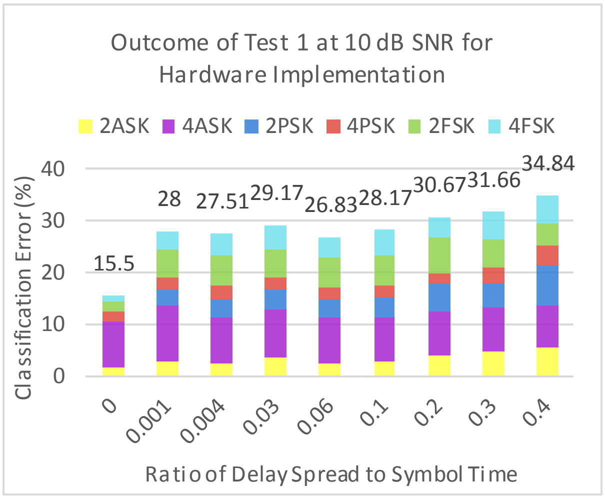


Figure 124: Classification Error for 400 training vectors under varying flat fading conditions at 10 dB SNR for Hardware Implementation

It can be seen that the misclassification of 2PSK was the biggest contributor to the classification error for 30 dB SNR and the misclassification of 4ASK for 10 dB SNR. It can also be seen that the classification error for 10 dB increased only a little as the fading ratio increased. For both 10 dB and 30 dB SNR it can again be seen that 4PSK is the most robust for classification in these channel conditions.

It can be concluded that the number of samples utilised for calculation of the feature values play a significant role in the classification performance of the decision tree. Potential remedies to this are discussed in the future work section 5.1.

4.4 Modulation Change Tracking

4.4.1 Implementation

The tracking was performed by logging the modulation output of the decision tree in a register to be utilised by external processes that may follow. The modulation output was recorded and evaluated after each classification. If the modulation type remained the same as the previous classification, no change was recorded. If the modulation type was different from the previous classification, a flag was set to indicate a change in modulation type and the status of the flag was recorded. The tracking process is illustrated in Figure 125.

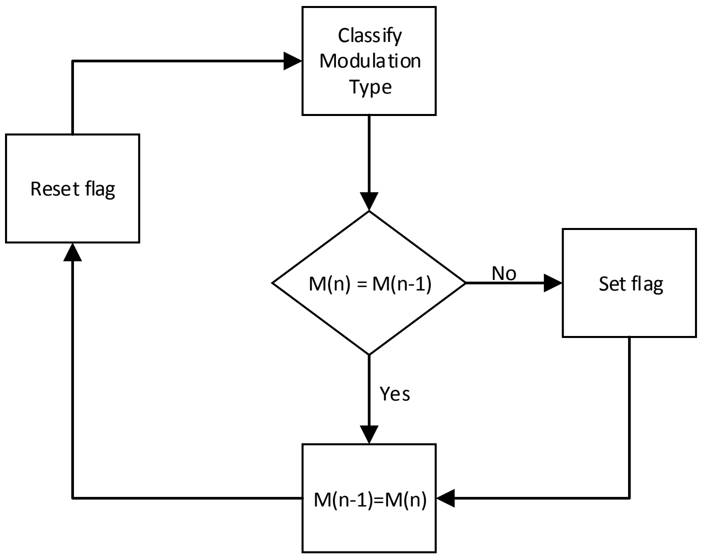


Figure 125: Flow Diagram for Tracking Changes in Modulation Types

4.4.2 Results

The logging of the modulation type sufficiently addressed the tracking from one modulation type to another. For deeper investigation into the tracking behaviour, the effect of modulation type transitions was investigated.

4.4.2.1 Modulation Transition Effects

In section 4.3.2.2 the classification of modulation types was performed over a window consisting of 2048 samples. Up to this point it was assumed that one type of modulation filled the whole window. The transition from one modulation type to another may however have an effect on the classification performance since varying signal lengths from the two different modulation types is contained within a single classification window. The next experiment was performed to investigate the effect of modulation transition within a window. The signals were investigated at an SNR of 30 dB. Two assumptions were made for this experiment. The first assumption is that data throughput wants to be maintained and a modulation type will therefore only change to another type of modulation with the same order. The second assumption is that there is no period during the transition from one modulation type to another in which a signal is not present. In other words, the modulation change is instantaneous. Practically there may be signal effects due to handover, and negotiation elements within the communication protocols. The experiment was performed by replacing samples of a modulation type incrementally with samples of another modulation type until the initial modulation type is absent and only the new modulation type was present. For each increment 10% of the initial modulation type is replaced with the other modulation type. For the hardware implementation 2048 samples were used for calculation of feature values, which is discussed in section 4.3.1. For this experiment 2050 samples where used to aid in the analysis process, since 2050 is divisible by 10 while 2048 is not. The difference in classification performance between 2048 and 2050 is assumed negligible. For each increment 500 iterations were performed. Since the features extracted from the signals are not sensitive to the time order in which they appear, inversing the order of modulation type switching would produce the same results. The decision tree constructed for 2048 samples in section 4.3.2.2 was used for this experiment. The outcome of the experiment is shown in Figure 126 to Figure 131.

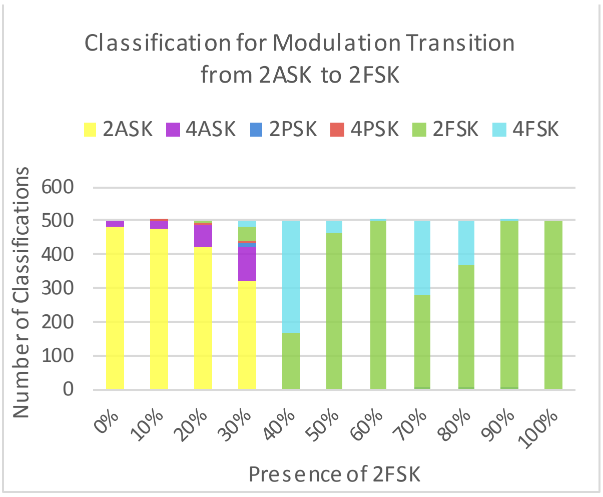


Figure 126: The Effect of Modulation Transition from 2ASK to 2FSK

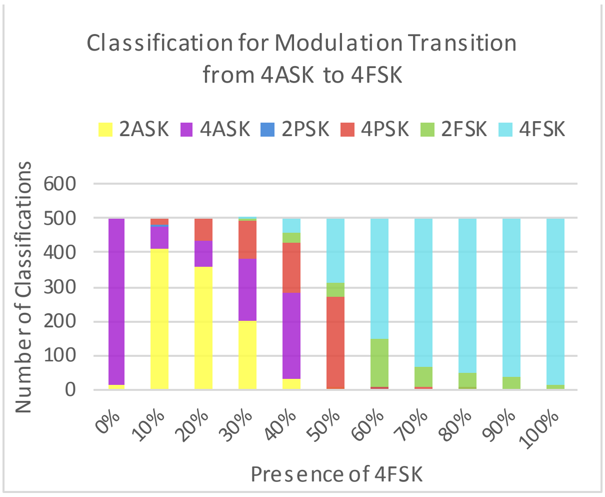


Figure 127: The Effect of Modulation Transition from 4ASK to 4FSK

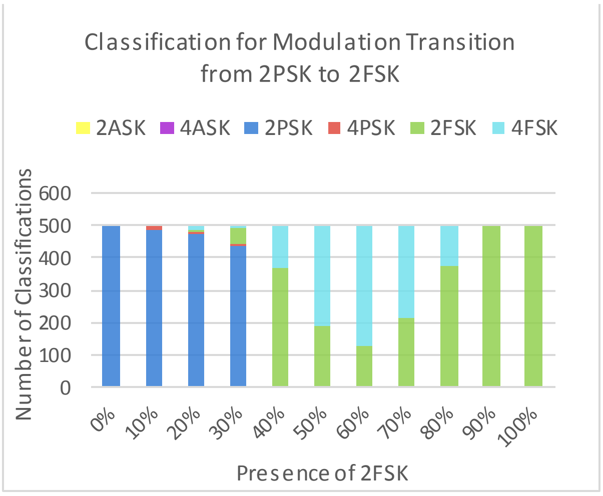


Figure 128: The Effect of Modulation Transition from 2PSK to 2FSK

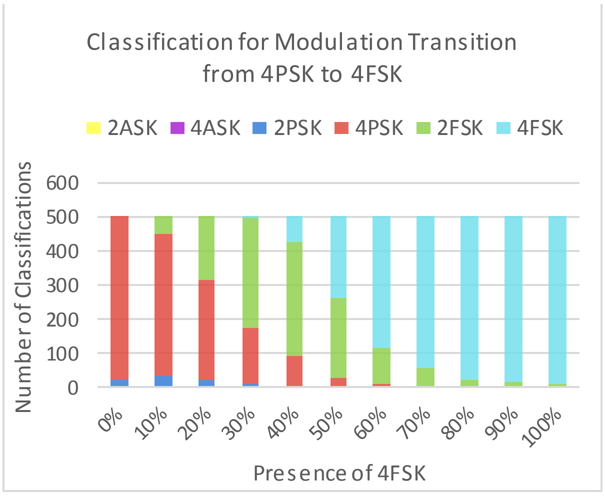


Figure 129: The Effect of Modulation Transition from 4PSK to 4FSK

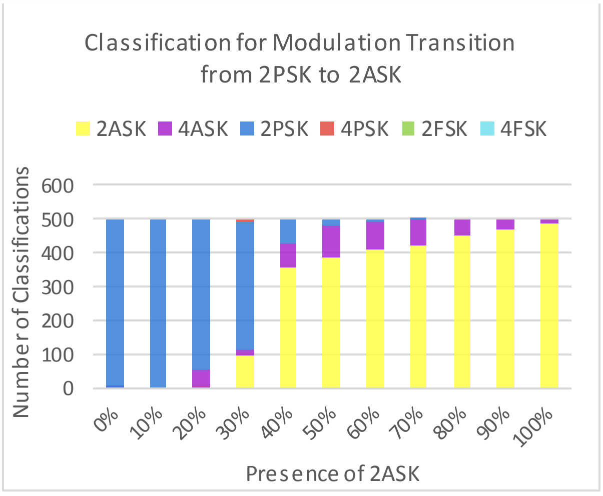


Figure 130: The Effect of Modulation Transition from 2PSK to 2ASK

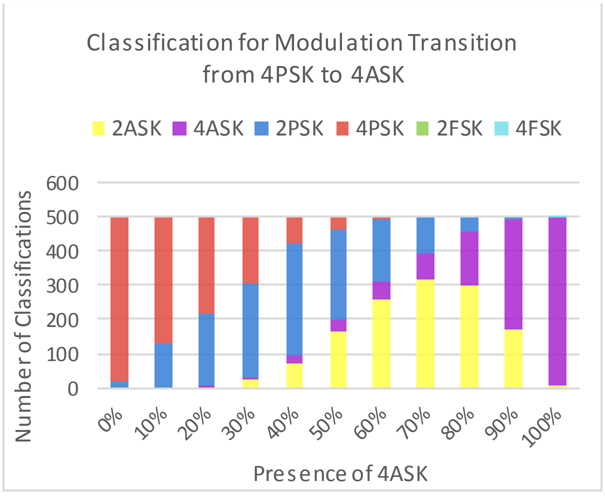


Figure 131: The Effect of Modulation Transition from 4PSK to 4ASK

From the results it can be seen that misclassifications occurred between different orders within modulations of the same type as both modulation types become equally present in the window. More specifically, the higher order modulation types were usually misclassified as the lower order modulation types. This misclassification was also evident in section 4.3.2.2. For modulation order of 2, 2FSK was the most affected and misclassified as 4FSK. While 2PSK was the least affected when dominantly present, 2ASK was the least affected when both modulation types were equally present and when least present in the window. For modulation order of 4, 4FSK was the least affected. 4ASK was the most affected, especially where 4ASK was less than 90% present. Although misclassifications of modulation orders of the same type occurred, it can be seen that very few misclassifications of other modulation types occurred.

4.5 Parameter Assumption Validity

For this study, the carrier frequency and signal bandwidth was assumed to be known, as is standard practice in literature to date [113], [110], [123], [69], [128]. However, when the carrier frequency and bandwidth is estimated in practice, uncertainties are introduced that could introduce offsets between the estimated values and the true values. These offsets could negatively affect the feature values which in turn could negatively affect classification accuracy.

The authors of [110] presented three methods to estimate the carrier frequency, although a preselected carrier frequency with zero estimation error was used in their work. Three methods were tested in simulation by the authors, in order to obtain the most accurate method for estimation of the carrier frequency. The tests were performed for an SNR of 5 dB, 10 dB, 15 dB and 20 dB. The results from [110] showed that their third method, modified zero-crossings, had 100% accurate estimation for an SNR above 10dB. For both 5 dB and 10 dB only 2FSK and 4FSK had an offset of 149.5$\pm$5.0 kHz and 149.1$\pm$2.7 kHz respectively. It can thus be seen the estimation of the carrier frequency becomes less accurate at lower SNR and may have an effect on feature values. This effect and the true performance of carrier frequency estimation techniques in hardware require further investigation in future work before conclusions can be drawn on the performance of the feature based classification method in practice.

In literature signals are matched to their bandwidth. This bandwidth varies for ASK, FSK and PSK (as can be seen in Table 1. This is knowledge that has to be estimated in a non-cooperative environment, or for which a design decision (i.e. fixed bandwidth) has to be made. Non-cooperative receivers normally have a bandwidth that is suited for various types of signals and is therefore wider to accommodate all the intended signals as opposed to cooperative receivers where the received signal parameters are known and filters can be matched exactly to those parameters. Since the amount of noise increases with the increase of a filter bandwidth, signals with narrower bandwidths will have more noise when filtered over a wider bandwidth. Features that are sensitive to noise may thus be affected and impact the feature values.

To investigate whether this wider noise filter and thus increased noise in the system impacts the feature values, noise with different bandwidths was simulated. The feature values of signals with noise filter bandwidths matched to the modulated signal, as described in section 4.1 and [110], was compared to signals with a fixed noise filter bandwidth. The bandwidth for the noise was chosen as

1.2 times the bandwidth selected for MFSK, which had the widest bandwidth of all the modulation types considered. Two features, σ_{af} and μ_{42}^{f}, showed notably different values. The differences in the feature values can be seen in Figure 132 to Figure 135.

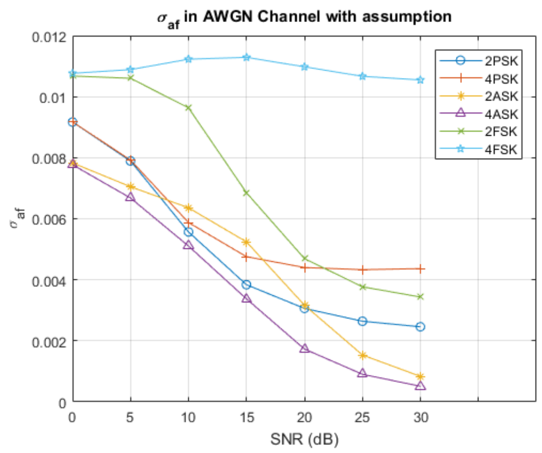


Figure 132: Feature values of σ_{af} in an AWGN Channel with assumption that signal bandwidth is known

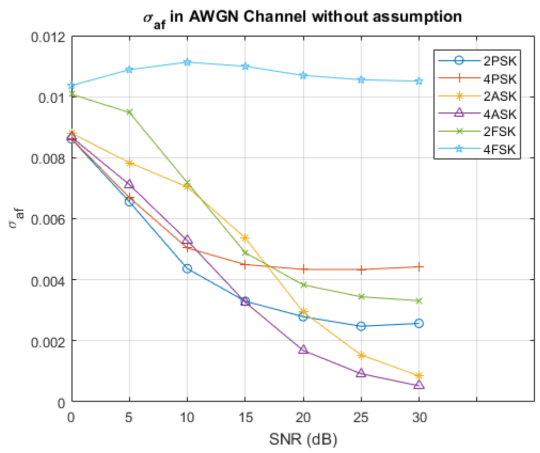


Figure 133: Feature values of σ_{af} in an AWGN Channel without assumption that signal bandwidth is known

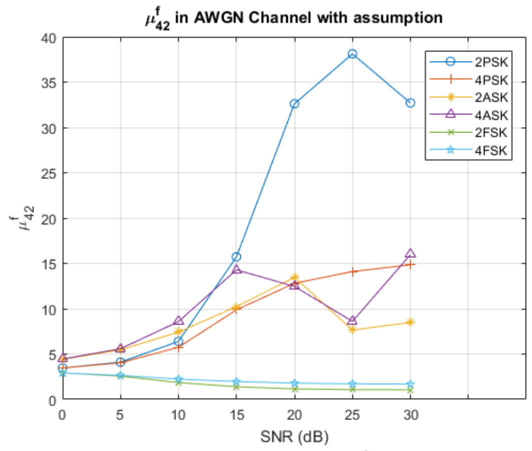


Figure 134: Feature values of μ_{42}^{f} in an AWGN Channel with assumption that signal bandwidth is correctly estimated

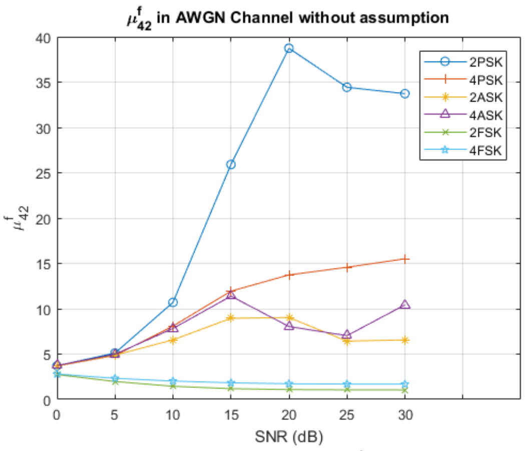


Figure 135: Feature values of μ_{42}^{f} in an AWGN Channel without assumption that signal bandwidth is correctly estimated

The differences thus required further investigation and the feature values were calculated only for the bandlimited noise thereafter. The feature values for the bandlimited noise generated according to the related modulated signals are shown in Appendix A.4. The plots were scaled to the values of the plots for signals in an AWGN channel, Figure 45 to Figure 52, to aid in comparison.

From the results in Appendix A.4 it can be seen that the features based on the instantaneous frequency were affected by the amount of noise due to the size of the bandwidth. The values of both features were different for different bandwidths. These features are thus sensitive to the bandwidth choice. Further investigation is required; it is however beyond the scope of this study. It is nevertheless worth noting that the feature values affected by the noise could potentially be used to derive valuable information such as the amount of noise in a system when σ_{af} and μ_{42}^{f} are observed. The SNR could possibly also be determined if the modulation type is known.

4.6 Conclusion

The design of the of the feature extraction, classification and tracking modules were implemented in this chapter. The design was first implemented in software by simulating an algorithm in Matlab. Several experiments were then performed to guide the design of the system and evaluate the performance of the design. The calculated instantaneous amplitude, phase and frequency were first compared to the theoretical instantaneous amplitude, phase and frequency from literature. It was determined that the instantaneous amplitude of MPSK contained weak intervals where symbol transitions occur. The transition effects influenced the values of the features and thus needed to be compensated for. Fluctuations in the instantaneous frequency of MPSK were also observed where symbol transitions occur. The calculation of the derivative of the phase to obtain the frequency caused unwanted fluctuations that were not observed in the theoretical representation in literature. These fluctuations were also compensated for by evaluating the values against a threshold and replacing any value that exceeded the threshold with a constant value.

The calculated feature values were compared to values obtained in [110] thereafter, since the signal parameters and features were adopted from their work. It was found that although the values differed from the values obtained in [110], there was a large degree of similarity in the values. The modulation types that have information in the instantaneous amplitude, phase and frequency had corresponding values for the associated features. The results thus still gave promising results to separate the different modulation types.

The feature values were analysed under varying SNR conditions. The SNR ranged from 0 dB to 30 dB in increments of 5 dB. A_{mean} and σ_a had very similar results although the values differed. It was also observed that there is a relation between A_{mean} and σ_a and μ_{42}^a. The phase based features, σ_{ap} and σ_{dp}, of MFSK and 4PSK were the most robust against noise conditions. 4ASK was more robust against noise when considering the amplitude based features. Similarly, 4FSK was more robust against noise when considering the frequency based features. From these results it was concluded that all modulation types were distinguishable from each other when the features were used in combination.

The features values were analysed under combinations of varying SNR and flat fading conditions. A static flat fading channel with three multi paths were simulated for the experiment. The features were tested against 9 different fading values that ranged from 0 to 0.4. A_{mean} and σ_a again showed very similar results although the values differed. It could also be observed that there was a relation between A_{mean} and σ_a and μ_{42}^a. σ_{ap} and σ_{dp} of MFSK and 4PSK were more robust against flat fading conditions. The amplitude based features of MASK were also more robust against flat fading as well as the frequency based features of 4FSK. It was also concluded that all modulation types were distinguishable from each other within the SNR and fading condition ranges, given the set of features evaluated.

Next the sensitivity of the algorithm to varying signal lengths was investigated. Signals with SNR and fading effects were used for the experiment. The minimum and maximum values among all the fading conditions were plotted for both 10 dB and 30 dB SNR. From the results it could also be seen that the values of different modulation types overlapped for some features and thus unambiguous results cannot be obtained with signals of different lengths. The results of this experiment were also used to determine whether the number of samples to use for hardware implementation was enough for accurate results.

A decision tree was constructed next to perform the classification of signal modulation types. The feature values obtained from the previously discussed experiments were used to train and test the decision tree. The classification accuracy of the tree was determined by using test vectors from the varying SNR dataset, the varying fading at 30 dB SNR dataset and the varying fading at 10 dB SNR dataset. The tree was also tested for decreasing training datasets. For the varying SNR the classification accuracy was very high and 100% classification accuracy was achieved for most tests of SNR greater than 0 dB. None of the tests achieved 100% classification accuracy for an SNR of 0 dB. Classification accuracy greater than 93% was however achieved for all the tests and a decrease in accuracy of 3% is observed between the biggest and smallest training dataset utilised. For varying fading at 30 dB SNR classification accuracies higher than 91% were achieved for all tests up to a fading ratio of 0.3 and higher than 87% for a fading ratio of 0.4. The largest decrease in accuracy between the biggest and smallest training dataset utilised, was smaller than 2% and occurred at a fading ratio of 0.4. For varying fading at 10 dB SNR classification accuracies higher than 79% were achieved for all tests under all the flat fading conditions. The largest decrease in classification accuracy between the biggest and smallest training dataset utilised, was smaller than 6% and occurred at a fading ratio of 0.06. It can be concluded that the constructed decision tree had good performance for various datasets containing feature values from signals experiencing different channel effects.

After these experiments were performed for the software simulation of the algorithm, a firmware design was implemented to an FPGA. A front-end processor was provided by the CSIR and was not implemented for this study. The effects of the processing performed by the front-end processor was however analysed to determine whether the effects of this processing could be considered negligible. It was found that the effect of the Hilbert filter and the effect of quantisation noise were negligibly small in comparison to the SNR of the signals tested, and was therefore ignored. The provided front-end processor also supplies the instantaneous amplitude, phase and frequency samples. It was therefore only necessary to implement the calculation of the features based on the instantaneous information on the hardware platform. The conversion of the I&Q signal samples to polar form of the front-end processor was analysed and the instantaneous amplitude, phase and frequency samples were generated accordingly using the same signals generated for the software simulation experiments. 2048 samples were used for the calculation of the feature values in hardware. It was found that the most feature values were similar to the results obtained in software, however the differences of some feature values can be ascribed to the difference in the number of samples used for calculations.

The feature values calculated in hardware were used to determine the classification accuracy of the decision tree when using the hardware results. It was found that the decision tree trained on longer signal lengths in software did not translate to the same level of accuracy for the shorter signal lengths. A new decision tree constructed from feature values calculated from 2048 samples was trained instead. The classification performance of the new tree was, as expected, lower than the tree constructed using longer signals. Classification accuracy higher than 73% and 57% was achieved for all training datasets for 30 dB and 10 dB SNR respectively. For a full training dataset, classification accuracy higher than 74% and 65% was achieved for 30 dB and 10 dB SNR respectively. Classification accuracy higher than 83% was achieved for all training datasets up to a fading ratio of 0.2 for 30 dB SNR. For 10 dB SNR, classification accuracy higher than 70% was achieved for al training datasets up to a fading ratio of 0.1. It can be seen that the classification accuracy decreased notably for fading ratios above 0.3.

It can be concluded that although the decision tree is adequate for signals experiencing various channel effects, the tree showed significantly reduced classification performance for signals with different signal lengths to what the tree was trained on. and that another tree had to be constructed. It was also concluded that 4PSK puts a limitation on the minimum number of samples required for accurate classification. An analysis should be performed to determine the optimal classification accuracy against the number of samples, which may vary from one application to another.

The modulation class output of the decision tree was used to track changes between modulation types by logging the modulation type over time. The effect of modulation transition within a classification window was investigated and it was found that that most misclassifications occurred between different orders of modulation types from the same family as 2 modulation types become equally present in the window. Although misclassifications between modulations of different orders of the same family occurred, very few misclassifications of other modulation types occurred. The modulation type that was most present in this window was most often correctly classified. It can be concluded that modulation types can at least be correctly classified as from the same family, when two modulation types are equally present in a classification window for most occurrences. Further research is required to study the full effect of modulation transitions on follow on processes that utilise this information, as well as to determine if a modulation transition can be detected and flagged within a single window, instead of using the difference of modulation type between two subsequent windows.

The similarity in the results of A_{mean} and σ_a and also the relation between A_{mean} and σ_a and μ_{42}^{a} raise the concern that there may be too many and thus irrelevant amplitude based features. More features may be needed that are not based on the instantaneous amplitude and rather based on the instantaneous phase or frequency might be needed.

5 CONCLUSION

5.1 Summary of Work

Automatic modulation classification is a challenging task in a non-cooperative environment where channel state information and signal parameters are not always available. Non-cooperative transmissions in military environments may be hampering or threatening to a user's own goals. In this environment signals can use never before seen modulation types or even modulation types that are specifically designed to avoid interception, detection and classification. Modulation is in effect used here as another layer of encryption. Modulation types thus have to be classified blindly, that is, without the use of a priori signal and channel state information. Adaptive modulation techniques complicate the task of classifying adversaries' signals even more, because the signal modulation changes quickly with time. It is desirable to be able to track the changes in an adversary emitter's modulation type. When the change from one modulation type to another modulation type in the signals from a transmitter can be tracked, the transmitter may be identified or their messages may be recovered, which is a critical aid in supporting battlefield decision making.

This study investigated the development of methods to classify transmitter modulation types, to aid in tracking changes in these modulation types in non-cooperative environments where adversaries use adaptive modulation techniques.

A complete capability required to track changes in transmitter modulation types includes the ability to receive and digitise signals of interest, spectrum sensing functionality to detect signals of interest, signal parameter estimation, classification of signal modulation, and the ability to track changes in that modulation. This study however only focused on the latter two steps, namely on developing an algorithm capable of tracking changes in modulation types through classification of signal modulation without the use of a priori signal information. Communication signals with modulation types Amplitude Shifts Keying (ASK) of order two and four, Phase Shift Keying (PSK) of order two and four, and Frequency Shift Keying (FSK) of order two and four were considered. The channel effects that were considered were AWGN noise and flat fading in a static multipath Rayleigh fading channel.

From literature it was found that there are two main approaches to classify and track modulation types in a non-cooperative environment. The first approach is Likelihood based classification, which formulates the classification as a composite hypothesis- testing problem. For this approach, each modulation type is assigned to the incoming signal under a hypothesis. The likelihood function is then used to find the correct modulation type of the signal. The second approach is feature based classification which entails 2 steps, feature extraction and decision making. Several features are extracted from the incoming signal and a decision is made based on the feature values.

From the literature study in Chapter 2 it was found that likelihood based classifiers are more accurate than feature based classifiers at the expense of computational complexity. The computations are repeated for each modulation hypothesis and each sample. Perfect channel knowledge is furthermore needed. For some LB methods one or two channel parameters can be unknown. The Expectation maximisation–maximum likelihood classifier is suitable in a non-cooperative environment, but is not cost effective in terms of computational complexity.

The main focus was to operate in a non-cooperative environment where many signal- and channel parameters may be unknown. A classifier that needs perfect channel knowledge becomes inoperable here. Secondly, the classification algorithm should be feasible for hardware implementation and the system should operate as quickly as possible with good classification accuracy. A classifier that is costly in terms of time and computational resources is thus undesirable. Computational complexity may also impose limitations for hardware implementation. Furthermore, in order to accurately track changes in modulation type of a signal in a non-cooperative environment, a classifier that is able to classify a wide variety of modulation types is needed.

Because operation in non-cooperative environment and computational cost took precedence in this study, feature based classifiers were considered. Feature based classification methods include the extraction of features based on the instantaneous amplitude, phase and frequency, features based on the wavelet transform and features based on higher order moments and cumulants of a signal. Features based on the instantaneous amplitude, phase and frequency were selected, because features can be extracted quickly without high computational complexity and is feasible for hardware implementation. Additionally, this method is able to operate in a non-cooperative environment and has the ability to classify a wide variety of modulation types.

For classification, several machine learning techniques were also investigated. From the three types of learning categories, namely supervised learning, unsupervised learning and reinforcement learning, supervised learning was selected since the objective is to classify, and data with labels were available. The most used machine learning techniques for feature based classification include Decision trees, Artificial Neural Networks (ANN), Support Vector Machines (SVM) and K-Nearest Neighbour (KNN) [91], [92].

Decision trees were chosen over ANN and SVM because they are fast to learn and predict. Their robustness against outliers and ability to select features in the training process are also great advantages over the other machine learning techniques. If necessary, decision trees can be used in ensemble through techniques such as random forest and random trees to improve performance by addressing the local optimum and high variance problems.

Based on the information gathered through the literature study, a conceptual design was completed. To perform tracking of modulation type changes, a system requires three main processes. The first step is to receive RF signals from the environment. The second step is to perform pre-processing on the received signals. After the necessary steps were taken to obtain the signal of interest and get it in its correct form, the classification of the modulation type and tracking of changes can take place, which is the third and final step. This study's main focus was on this last step, within the context of the first two. An understanding of the previous steps was however required in order to design the third step correctly. A conceptual design encapsulating all three steps was performed.

From literature, eight general features were identified. During this investigation, a feature with high computational complexity was exchanged with a similar yet less computationally expensive one. The design of the feature extraction, classification and tracking was first modelled in the software modelling tool Matlab, whereafter a part of the design was translated to hardware using Xilinx's Vivado 2016.2 environment. Several experiments were performed to evaluate the performance of the design. It was found that symbol transitions of MPSK had effects on the calculation of the instantaneous amplitude and frequency. These effects had to be compensated for by means of a threshold method, to ensure that the feature values calculated were not adversely affected.

The feature values were analysed under varying SNR and flat fading channel conditions. For varying SNR the phase based features, σ_{ap} and σ_{dp}, showed the most robust behaviour when calculated on signals consisting of modulation types of MFSK and 4PSK. 4ASK was also more robust against noise when only considering the amplitude based features. Similarly, 4FSK was more robust against noise when considering only the frequency based features. For varying SNR and fading conditions, a static flat fading channel with three multi paths at 30 dB and 10 dB SNR was considered. Features σ_{ap} and σ_{dp} showed the most robust behaviour when calculated on signals consisting of modulation types of MFSK and 4PSK. Amplitude based features when calculated on MASK, and frequency based features when calculated on 4FSK was also more robust than the rest of the modulation types when calculated in the presence of flat fading.

An important variable in this study was the signal length. Shorter signals consume less hardware resources and reduce calculation time. Additionally, since transmitters in a non-cooperative environment can change their modulation type quickly, it is also desirable to classify modulation types using the fewest samples possible for quicker classification turnaround time. An analysis on the selected set of features over the entire range of SNR and fading conditions considered for this study showed that feature values of different modulation types overlapped. However, most feature values remained separable for signals with more than 1920 samples.

A decision tree was constructed to perform the classification of signal modulation types. A dataset of simulated random signals modulated by the 6 chosen modulation types were generated both with bandlimited white noise only, and bandlimited white noise with flat fading. The dataset contained signals with an SNR from 0 dB to 30 dB as well as flat fading ratios from 0 to 0.4. This dataset was used to train and test the decision tree. Results showed that the classification performance was insensitive to SNR, and achieved perfect prediction performance for SNR values greater than 5 dB. None of the tests achieved 100% classification accuracy for an SNR of 0 dB. Classification accuracy higher than 93% was however achieved for all the tests, and a decrease of 3% is observed between the utilisation of the biggest training dataset and the smallest training dataset. At 10 dB SNR under varying fading conditions classification accuracies higher than 91% were achieved for all tests up to a fading ratio of 0.3 and higher than 87% for a fading ratio of 0.4. The largest decrease in accuracy between the biggest and the smallest training dataset utilised was smaller than 2% and occurred at a fading ratio of 0.4. For varying fading at 10 dB SNR classification accuracies higher than 79% were achieved for all tests under all the flat fading conditions. The largest decrease in classification accuracy between the biggest and the smallest training dataset utilised was smaller than 6% and occurred at a fading ratio of 0.06. Thus the decision tree can be trained on a small amount of data, however this warrants further investigation.

For the hardware implementation, the feature extraction step was implemented on the FPGA on the hardware platform, as subsequent steps are better suited to DSPs or CPUs , that do not form part of the hardware platform at this time. A front-end processor was provided which supplies the instantaneous amplitude, phase and frequency samples. The front-end processor was however not used for the lab based hardware tests, since this platform is designed for high, agile bandwidth signals and operates at a sample frequency of 4GHz. A great amount of memory is required to generate signals for the front-end processor at the full bandwidth, whereafter a filter and decimation significantly reduces the data rate. This method for testing is prohibitive and a better simulation approach was thus required. The effect of the front end processing within the frequency band of interest was therefore analysed instead to determine whether it introduces significant effects that should be taken into consideration. No significant effects were found and I&Q samples where generated accordingly in Matlab. The feature extraction block was designed to match the interface specifications of the front-end processor with the aim of future integration. A sample length of 2048 was used for the calculation of the feature values in hardware. It was found that the most feature values were similar to the results obtained in software, however some differences were observed that is attributed to the difference in the sample lengths used.

The feature values calculated in hardware were used to determine the classification accuracy of the decision tree when using the hardware results. It was found that although the decision tree was adequate for signals experiencing various channel effects, the tree showed significantly reduced classification performance for signals with different signal lengths to what the tree was trained on. To overcome this limitation, a new tree had to be constructed. The classification performance of the new tree was, as expected, lower than the tree constructed using longer signals, performing 7.9% and 21.2% worse on average for the biggest dataset at an SNR of 30 dB and 10 dB respectively, which were 14.99% and 8.04% more than the initial tree.

The modulation class output of the decision tree was used to track changes between modulation types by logging the modulation type over time. The effect of modulation transition within a classification window was investigated and it was found that that most misclassifications occurred between different orders of modulation types from the same family as 2 modulation types become equally present in the window. For modulation order of 2, 2FSK was the most affected and misclassified as 4FSK. For modulation order of 4, 4FSK was the least affected. 4ASK was the most affected, and misclassified as 2ASK. Although misclassifications between modulations of different orders of the same family occurred, very few misclassifications of other modulation types occurred. The modulation type that was most present in this window was most often correctly classified.

Considering the results obtained in this study, many valuable conclusions can be drawn. It was shown in literature that the classification accuracy by using features based on the instantaneous amplitude, phase and frequency for varying SNR conditions is high in the context of the assumptions [69], [103], [113], which was confirmed in this study. The classification accuracy on these features in varying fading conditions has received little investigation in literature up to this point. The results obtained in this study have shown that these features can be used in flat fading channels under varying SNR conditions, with only a slight reduction in performance of 2.74% and 6.85% on average for an SNR of 30 dB and 10 dB respectively. The calculation of these features in hardware was published once before [129]. In our study it was shown that these features are feasible for hardware implementation and that the results can be used for classification by means of a decision tree. When two different modulation types were present in a classification window, it was shown that the signal was classified as the same modulation type as the signal that is most present in the window in most cases. Misclassifications between modulation types from the same family with different orders did however occur. It was shown that further investigation is required to determine the number of samples required for feature calculation to achiever high classification accuracy, and the assumptions of known signal centre frequency and accurate noise filter bandwidth has to be challenged as these could have serious consequences for implementation.

5.2 Outcome of Study

The objective of this study was to classify and track changes of modulation types from a communications transmitter in a non-cooperative environment without channel state information. The secondary objective was to develop the method in such a way that the digital signal processing components thereof can be implemented on a hardware platform provided by the CSIR.

During this study, feature based classification was used to successfully classify modulation types of signals from a single communications transmitter without the use of channel state information. This was achieved by using features based on the instantaneous amplitude, phase and frequency of a signal for feature extraction and a decision tree for classification. The method was tested under varying SNR conditions from 0 dB to 30 dB and performed well, achieving classification accuracy higher than 96 % for the worst SNR condition. The change from one modulation type to another was successfully performed by logging the modulation type, and any changes, over time for use by external processes. The secondary objective was successfully achieved by implementing the feature extraction process on a hardware demonstrator provided by the CSIR. The feature values obtained in hardware reflected the results obtained in software. The feature values obtained in hardware was also successfully used to classify the proposed modulation types. The study thus successfully addressed all of the research objectives.

Additionally to the objectives of the study, many other goals were achieved. The classification was successfully performed under varying flat fading ratios from 0 to 0.4 in a static multipath Rayleigh fading channel at an SNR of 30 dB and 10 dB. Classification accuracy higher than 89% and 84% were achieved for the worst fading condition at 30 dB and 10 dB SNR respectively. In addition to the investigation under varying fading conditions, the effect of signal length and training dataset size were also investigated. Furthermore, a tree was reconstructed to improve the classification performance for feature values obtained from hardware. Lastly the effect of modulation transitions within a classification window was also investigated, which was beyond the scope of this study.

Many valuable discoveries were also made during the investigation of the additional work. It was found that signal length has a significant effect on the classification performance. It was found that some of the feature values had correspondence in their results and there may be a large degree of duplicate information between the features. This discovery led to the exploring of a new feature which yielded promising results. The new feature gave insight and raised many questions on how new, less complex features can be utilised without reducing classification accuracy while reducing computational load.

5.3 Future work

Some challenges were identified during this study. The similarity in the results of A_{mean} and σ_a and also the relation between A_{mean} and σ_a and μ_{42}^{a} raised the concern that there may be a large degree of duplicate information between the amplitude features, that could simply increase processing load and not yield significant classification accuracy gains. Additionally, more unique features which are not based on the instantaneous amplitude and rather based on the instantaneous phase or frequency might be needed. It was beyond the scope of this study to derive and investigate the utilisation of new features. The utilisation of the standard deviation of the direct value of the phase, σ_{df}, was however investigated, since the standard deviation of the direct value of both the phase and amplitude were already calculated and the information was thus readily available to use for calculation of the feature with minimal additional effort. This feature was not found in literature and from Figure 136 to Figure 145, it can be seen that this new feature shows promise, subject to further investigation.

With this new feature, 2FSK and 4FSK remain separable from the other modulation types for all the proposed channel conditions. When comparing σ_{df} based on the direct value with its absolute value counterpart σ_{af} which is used in literature, it is seen that σ_{df} provides additional separability for 2FSK where σ_{af} did not.

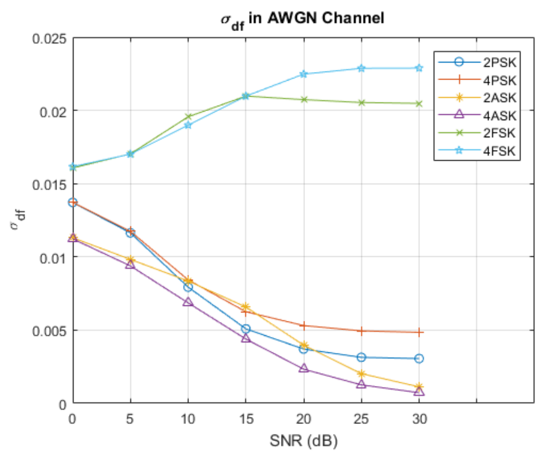


Figure 136: Feature values of σ_{df} in an AWGN Channel in Matlab Simulation

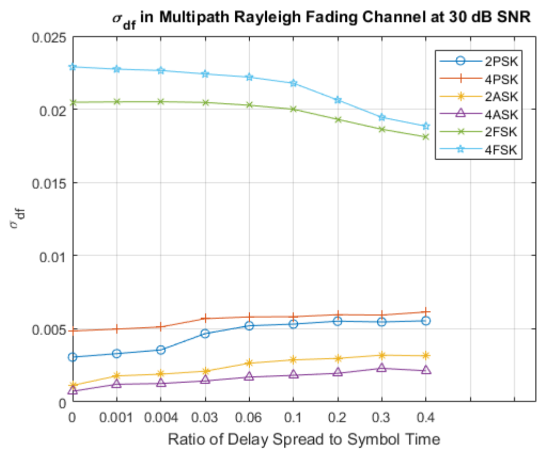


Figure 137: Feature values of σ_{df} in a Flat Fading Channel at 30 dB SNR in Matlab Simulation

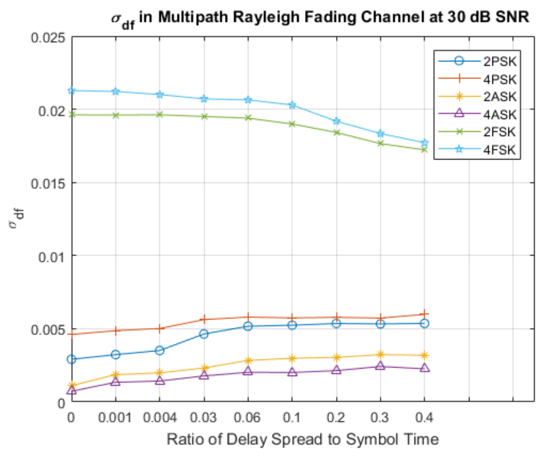


Figure 138: Feature values of σ_{df} in a Flat Fading Channel at 30 dB SNR for Hardware Implementation

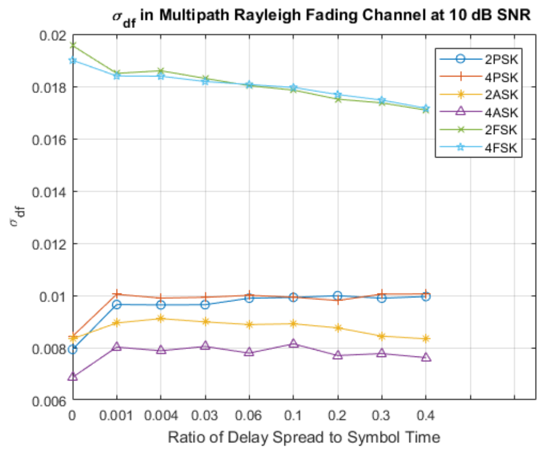


Figure 139: Feature values of σ_{df} in a Flat Fading Channel at an SNR of 10 dB in Matlab Simulation

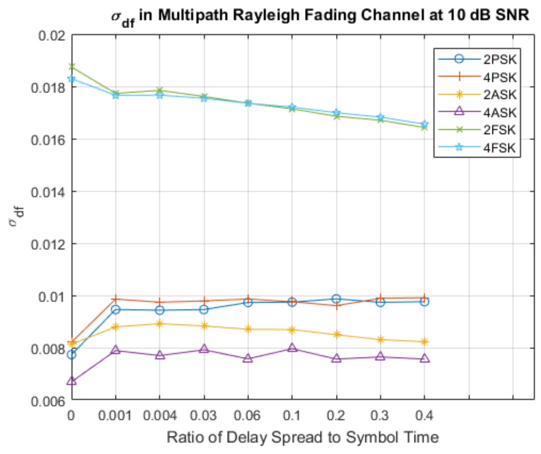


Figure 140: Feature values of σ_{df} in a Flat Fading Channel at an SNR of 10 dB for Hardware

The results of σ_{df} raise many questions. Since σ_a and A_{mean} are very similar, will σ_{dp} and σ_{df} also be similar to the mean of the absolute value of the phase and frequency respectively? The calculation of the kurtosis occupies much more device resources, is it possible to replace the kurtosis with another feature without reducing the classification accuracy? Can more modulation types be classified with new features? Can one simply or create other, less complex features that do not reduce classification accuracy while reducing computational load.

Although a wide range of modulation types can be classified using features based on the instantaneous amplitude, phase and frequency, this study was limited to six digital modulation types. The decision tree was trained to classify these six modulation types exclusively. An obvious extension of this work is to include more modulation types, possibly guided by a study that shows the prevalence of modulation types in modern use. The combination of features to classify a wider range of modulation types such as a combination of higher order statistics to classify higher order modulation types as well, can be considered. An example of combining features to classify a wide range of modulation types can be found in [100].

In Chapter 1, unknown signals that might be specifically designed to avoid sensing, detection and classification were identified as one of the challenges in a non-cooperative environment. Although a wide variety of modulation types can be classified and more features can be added in order to enlarge the modulation pool, these unknown signals might still be able to avoid classification. Unsupervised learning is a method capable of finding patterns in data and grouping data with similar characteristics together. Such methods can be used in collaboration with the current algorithm to identify these signals. The results of the unsupervised learning algorithms can be used as feedback to the current algorithm and these signals can then potentially be classified or characterised in this manner.

The sample frequency, 1200 kHz, of the signals from which features were calculated was much higher than required to prevent aliasing to occur. This sample frequency was however used in order to compare with the results in [110]. It is expected that different sample frequencies will have an effect on the feature values and that there will be a trade-off between the sample frequency and the degree of separability that could be achieved with the features. An investigation is thus required to determine the effect of the sample frequency on the feature values.

For this study features were only investigated in a static flat fading channel. Further research is required to determine whether the use of a threshold to find the weak intervals in the instantaneous amplitude and the fluctuations in the instantaneous frequency due to the symbol transitions of MPSK will be adequate in channels where signals might experience other fading effects than flat fading. Future work also includes determining how well the system works when tested against real world data.

The decision tree constructed with the feature values extracted in software was adequate for classifying signals experiencing various channel effects, however classification performance suffered when the signal length was varied. Most differences in feature values between software and hardware results were due to the utilisation of different signal lengths. Further research is thus required to determine the classification performance of decision trees when utilising different signal lengths and to determine the optimal number of samples needed to construct a tree that still exhibits good classification performance while minimising signal length. Further research is also required to determine how fast a transmitter could change its modulation type in order to determine if enough signal samples can be obtained for accurate classification as the minimum signal length is an obvious limitation in this scenario.

The modulation class output of the decision tree was used to track changes between modulation types by logging the modulation type over time and the effect of modulation transition within a window was investigated. Further research is required to study the full effect of modulation transitions on follow on processes that utilise this information, as well as to determine if a modulation transition can be detected and flagged within a single window, instead of using the difference of modulation type between two subsequent windows.

BIBLIOGRAPHY

[1] L. S. Cohen and R. J. Jost, “Spectrum Analysis and Measurements in a Congested Electromagnetic Environment,” in *IEEE International Symposium on Electromagnetic Compatibility (EMC)*, Denver, 2013.

[2] Z. Zhu and A. K. Nandi, “Introduction,” in *Automatic Modulation Classification- Principles, Algorithms and Applications*, Chichester, John Wiley & Sons Ltd, 2015, pp. 1-16.

[3] G. I. o. Technology, “Next-Generation Electronic Warfare Development Targets Fully Adaptive Threat Response Technology,” Georgia Tech Research Institute, [Online]. Available: http://www.gtri.gatech.edu/casestudy/GTRI-next-generation-electronic-warfare-angry-kitten. [Accessed 23 February 2016].

[4] H. Glaude, C. Enderli, J. Grandin and O. Pietquin, “Learning of Scanning Strategies for Electronic Support using Predictive State Representations,” in *IEEE International Workshop on Machine Learning for Signal Processing*, Boston, 2015.

[5] S. L. Kadambe, B. J. Haan, J. A. Fuemmeler, W. S. Spencer, C. J. Chavez and R. J. Frank, “Cognitive Networked Electronic Warfare”. United States of America Patent US8494464 B1, 23 July 2013.

[6] P. Shih and D.-C. Chang, “An Automatic Modulation Classification Technique Using High-Order Statistics for Multipath Fading Channels,” in *11th International Conference on ITS Telecommunications (ITST)*, St. Petersburg, 2011.

[7] M. Narendar, “Automatic Modulation Classification for Cognitive Radios using Cumulants based on Fractional Lower Order Statistics,” in *General Assembly and Scientific Symposium*, Istanbul, 2011.

[8] V. Iglasias, J. Grajal and O. Yeste-Ojeda, “Automatic Modulation Classification for Military Applications,” in *19th European Signal Processing Conference*, Barcelona, 2011.

[9] A. E. Spezio, “Electronic Warfare Systems,” *IEEE Transactions on Microwave Theory and Techniques,* vol. 50, no. 3, pp. 633-644, 2002.

[10] Z. Zhu and A. K. Nandi, “Applications of AMC,” in *Automatic Modulation Classification- Principles, Algorithms and Applications*, Chichester, John Wiley & Sons, Ltd, 2015, pp. 2-5.

[11] E. E. Azzouz and A. K. Nandi, “Automatic Modulation Recognition-I,” *Journal of the Franklin Institute,* vol. 334, no. 2, pp. 241-273, 1997.

[12] A. Elrharras, R. Saadane, M. Wahbi and A. Hamdoun, “Signal Detection and Automatic Modulation Classification Based Spectrum Sensing Using PCA-ANN with Real Word Signals,”

Applied Mathematical Science, vol. 8, no. 160, pp. 7959-7977, 2014.

[13] J. J. Popoola and R. van Olst, “Application of Neural Network for Sensing Primary Radio Signals in a Cognitive Radio Environment,” in *AFRICON 2011*, Livingston, 2011.

[14] L. Haring and C. Kisters, “Constraint-based Adaptive OFDM Transmission with Signaling-assisted Modulation Classification,” in *Asilomar Conference on Signals, Systems and Computers 2013*, California, 2013.

[15] A. Tsaknalis, S. Chatzinotas and B. Ottersten, “Automatic Modulation Classification for Adaptive Power Control in Cognitive Satellite Communications,” in *Advanced Satellite Multimedia Systems Conference and the 13th Signal Processing for Space Communications Workshop (ASMS/SPSC)*, Livorno, 2014.

[16] A. Tsakmalis, S. Chatzinotas and B. Ottersten, “Modulation and Coding Classification for Adaptive Power Control in 5G Cognitive Communications,” in *IEEE 15th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Toronto, 2014.

[17] D.-C. Chang and P.-K. Shih, “Cumulants-based Modulation Classification Technique in Multipath Fading Channels,” *IET Communications,* vol. 9, no. 6, pp. 828-835, 2015.

[18] D. Cabric and R. W. Brodersen, “Physical Layer Design Issues Unique to Cognitive Radio Systems,” in *IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications*, Berlin, 2005.

[19] Z. Zhu and A. K. Nandi, “Modulation Classification Design for Military Applications,” in *Automatic Modulation Classification- Principles, Algorithms and Applications*, Chichester, John WIley & Sons Ltd, 2015, pp. 153-160.

[20] D. F. C. Moura, F. A. B. da Silva and J. F. Galdino, “Case Studies of Attacks over Adaptive Modulation Based Tactical Software Defined Radios,” *Journal of Computer Networks and Communications,* vol. 2012, pp. 1-9, 2012.

[21] A. Al-Dulaimi, N. Radhi and H. S. Al-Raweshidy, “Cyclostationary Detection of Undefined Secondary Users,” in *Third International Conference on Next Generation Mobile Applications, Services and Technologies*, Cardiff, 2009.

[22] K. Olivier and M. Gouws, “Modern Wideband DRFM Architecture and Real-time DSP Capabilities for Radar Test and Evaluation,” in *Saudi International Electronics, Communications and Photonics Conference*, Fira, 2013.

[23] Z. Zhu and A. K. Nandi, “Signal Models for Modulation Classification,” in *Automatic Modulation Classification- Principles, Algorithms and Applications*, Chichester, John Wiley & Sons, Ltd, 2015, pp. 19-33.

[24] S. Haykin and M. Moher, “Propagation and Noise,” in *Modern Wireless Communications*, New

Jersey, Pearson Education Inc., 2005, pp. 11-102.

[25] L. E. Frenzel, “Radio Wave Propagation,” in *Principles of Electronic Communication Systems 4th Ed*, New York, Mc Graw Hill, 2014, pp. 538-551.

[26] A. Mitra, “Multipath wave propagation and fading,” in *Lecture notes on mobile communication*, Guwahati, Indian Institute of Technology Guwahati, 2009, pp. 75-100.

[27] Mathuranathan, “Power Delay Profile,” GaussianWaves, 9 July 2014. [Online]. Available: http://www.gaussianwaves.com/2014/07/power-delay-profile/. [Accessed 4 September 2017].

[28] M. Narendar, “Automatic Modulation Classification for Cognitive Radios using Cumulants based on Fractional Lower Order Statistics,” in *General Assembly and Scientific Symposium*, Istanbul, 2011.

[29] M. Abdelbar, B. Tranter and T. Bose, “Cooperative Modulation Classification of Multiple Signals in Cognitive Radio Networks,” in *IEEE International Conference on Communications (ICC)*, Sydney, 2014.

[30] M. Zaerin and B. Seyfe, “Multiuser Modulation Classification Based on Cumulants in Additive White Gaussian Noise Channel,” *IET Signal Processing*, vol. 6, no. 9, pp. 815-823, 2013.

[31] S. E. El-Khamy, H. A. Elsayed and M. M. Rizk, “Classification of Multi-User Chirp Modulation Signals Using Higher Order Cumulant Features and Four Types of Classifiers,” in *28th National Radio Science Conference (NRSC)*, Cairo, 2011.

[32] B. Ramkumar, T. Bose and M. S. Radenkovic, “Robust multiuser automatic modulation classifier for multipath fading channels,” in *IEEE Symposium on New Frontiers in Dynamic Spectrum*, Singapore, 2010.

[33] S. S. Sohul, B. Ramkumar and T. Bose, “Multiuser Automatic Modulation Classification for Cognitive Radios using Distributed Sensing in Multipath Fading Channels,” in *7th International ICST Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM)*, Stockholm, 2012.

[34] E. Like, V. Chakravarthy, R. Husnay and Z. Wu, “Modulation Recognition In Multipath Fading Channel Using Cyclic Spectral Analysis,” in *IEEE Global Telecommunications Conference*, New Orleans, 2008.

[35] E. Rebeiz and D. Cabric, “Blind Modulation Classification Based on Spectral Correlation and its Robustness to Timing Mismatch,” in *2011 Military Communications Conference*, Baltimore, 2011.

[36] X. Tan, H. Zhang, Y. Sheng and W. Lu, “Blind Modulation Recognition of PSK Signals Based on Constellation Reconstruction,” in *International Conference on Wireless Communications and Signal Processing (WCSP)*, Suzhou, 2010.

[37] G. J. Phukan and P. K. Bora, “An Algorithm to Mitigate Channel Distortion in Blind Modulation Classification,” in *National Conference on Communications (NCC)*, New Delhi, 2013.

[38] S. Majhi, P. B. J and M. Kumar, “Blind Wireless Receiver Performance for Single Carrier Systems,” in *International Conference on Communications and Signal Processing (ICCSP)*, Melmaruvathur, 2015.

[39] H. Ketterer, F. Jondral and A. H. Cost, “Classification of Modulation Modes using Time-frequency Methods,” in *IEEE International Conference Acoustics, Speech, and Signal Processing*, Phoenix, 1999.

[40] H. Agirman-Tosun, Y. Liu, A. M. Haimovich and O. Simeone, “Modulation Classification of MIMO-OFDM Signals by Independent Component Analysis and Support Vector Machines,” in *The Forty Fifth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, Pacific Grove, 2001.

[41] W. Guibene and D. Slock, “Signal Classification in Heterogeneous OFDM-based Cognitive Radio Systems,” in *20th International Conference on Telecommunications (ICT)*, Casablanca, 2013.

[42] M. L. D. Wong, S. K. Ting and A. K. Nandi, “Naïve Bayes Classification of Adaptive Broadband Wireless Modulation Schemes with Higher Orde Cumulants,” in *2nd International Conference on Signal Processing and Communication Systems*, Gold Coast, 2008.

[43] J. L. Xu, W. Su and M. Zhou, “Likelihood-Ratio Approaches to Automatic Modulation Classification,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews),* vol. 41, no. 4, pp. 455-469, 2011.

[44] D. Shimbo and I. Oka, “A Modulation Classification Using Amplitude Moments in OFDM Systems,” in *International Symposium on Information Theory and its Applications (ISITA)*, Taichung, 2010.

[45] S. H. O. Salih, A. Al-Refai, M. Suliman and A. Mohammed, “Implementation of Adaptive Modulation for Broadband,” in *International Conference on Computing, Electrical and Electronics Engineering (ICCEEE)*, 2013.

[46] L. Haring and C. Kisters, “MAP-based Automatic Modulation Classification for Wireless Adaptive OFDM Systems,” in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, 2013.

[47] L. Haring and C. Kisters, “Performance Comparison of Adaptive Modulation and Coding in OFDM Systems Using Signalling and Automatic Modulation Classification,” in *17th International OFDM Workshop 2012*, Essen, 2012.

[48] O. A. Dobre, A. Abdi, B.-N. Y and W. Su, “Selection combining for modulation recognition in fading channels,” in *Military Communications Conference (MILCOM)*, Atlantic City, 2005.

[49] O. A. Dobre, A. Abdi, B.-N. Yeheskel and S. Wei, "A Survey of Automatic Modulation Classification Techniques: Classical Approaches and New Trends," *IET Communications,* vol. 1, no. 2, pp. 137-156, 2007.

[50] O. A. Dobre, A. Abdi, Y. Bar-Ness and W. Su, "Blind modulation classification: a concept whose time has come," in *Sarnoff Symposium on Advances in Wired and Wireless Communication*, 2005.

[51] Z. Zhu and A. K. Nandi, "Likelihood-based Classifiers," in *Automatic Modulation CLassification: Principles, ALgorithms and Applications*, Chichester, John Wiley and Sons, 2015, pp. 35-48.

[52] V. G. Chavali and C. R. C. M. da Silva, "Maximum-Likelihood Classification of Digital Amplitude-Phase Modulated Signals in Flat Fading Non-Gaussian Channels," *IEEE Transactions on Communications,* vol. 59, no. 8, pp. 2051-2056, 2011.

[53] E. Soltanmohammadi and M. Naraghi-Pour, "Blind Modulation Classification over Fading Channels Using Expectation-Maximization," *IEEE Communications Letters,* vol. 17, no. 9, pp. 1692-1695, 2013.

[54] N. Alyaoui, H. B. Hnia, A. Kachouri and S. Mounir, "The Modulation Recognition Approaches for Software Radio," in *2nd International Conference on Signals, Circuits and Systems*, Monastir, 2008.

[55] Y. L. Chao and R. A. Scholtz, "Ultra-wideband Transmitted Reference Systems," *IEEE Transactions on Vehicular Technology,* vol. 54, no. 5, pp. 1556-1569, 2005.

[56] Z. Yonghong, Y. C. Liang and P. T. Hanh, "Spectrum Sensing for OFDM Signals Using Pilot Induced Auto-Correlations," *IEEE Journal on selected Areas in Communications,* vol. 31, no. 3, pp. 353-363, 2013.

[57] P. Wang, H. Li and B. Himed, "A New Parametric GLRT for Multichannel Adaptive Signal Detection," *IEEE Transactions on Signal Processing,* vol. 58, no. 1, pp. 317-325, 2010.

[58] J. Leinonen and M. Juntti, "Modulation Classification in Adaptive OFDM Systems," in *IEEE 59th Vehicular Technology Conference*, Milan, 2004.

[59] P. Panagiotou, A. Anastasopoulos and A. Polydoros, "Likelihood Ratio Tests for Modulation Classification," in *21st Century Military Communications Conference*, Los Angeles, 2000.

[60] O. A. Dobre and F. Hameed, "Likelihood-Based Algorithms for Linear Digital Modulation Classification in Fading Channels," in *Canadian Conference on Electrical and Computer Engineering*, Ottawa, 2006.

[61] M. Derakhitian, A. A. Tadaion and S. Gazor, "Modulation classification of linearly modulated signals in slow flat fading channels," *IET Signal Processing,* vol. 5, no. 5, pp. 443-450, 2011.

[62] F. Hameed, O. A. Dobre and D. C. Popescu, “On the likelihood-based approach to modulation classification,” *IEEE Transactions on Wireless Communications,* vol. 8, no. 12, pp. 5884-5892, 2009.

[63] Z. Zhu and A. K. Nandi, “Blind Modulation Classification,” in *Automatic Modulation Classification: Principles, Algorithms and Applications*, Chichester, John WIley and Sons, 2015, pp. 97-108.

[64] O. Ozdemir, T. Wimalajeewa, B. Dulek, P. K. Varshney and W. Su, “Asynchronous Linear Modulation Classification With Multiple Sensors via Generalized EM Algorithm,” *IEEE Transactions on Wireless Communications,* vol. 14, no. 11, pp. 6389-6400, 2015.

[65] Z. Zhu and A. K. Nandi, “Comparison of Modulation Classifiers,” in *Automatic Modulation CLassification: Principles, Algorithms and Applications*, Chichester, John Wiley and Sons, 2015, pp. 109-140.

[66] Z. Zhu and A. K. Nandi, “Modulation Classification Features,” in *Automatic Modulation CLassification: Principles, ALgorithms and Applications*, Chichester, John Wiley and Sons, 2015, pp. 65-80.

[67] E. E. Azzouz and A. K. Nandi, “Recognition of Analogue Modulations,” in *Automatic Modulation Recognition of Communication Signals*, Dordrecht, Kluwer Academic Publishers, 1996, pp. 42-76.

[68] J. J. Popoola and R. van Olst, “Automatic Classification of Combined Analog and Digital Modulation Schemes using Feedforward Neural Network,” in *AFRICON*, Livingstone, 2011.

[69] E. Moser, M. K. Moran, E. Hillen, D. Li and Z. Wu, “Automatic Modulation Classification via Instantaneous Features,” in *National Aerospace and Electronics Conference*, Dayton, 2015.

[70] Q. Zhou, H. Lu, L. Jia and M. Kefei, “Automatic Modulation Classification with Genetic Backpropagation Neural Network,” in *IEEE Congress on Evolutionary Computation*, Vancouver, 2016.

[71] K. C. Ho, W. Prokopiw and Y. T. Chan, “Modulation Identification of Digital Signals by the Wavelet Transform,” *Radar, Sonar and Navigation ,* vol. 147, no. 4, pp. 169-176, 2000.

[72] L. Hong and K. C. Ho, “Identification of Digital Modulation Types using the Wavelet Transform,” in *IEEE Military Communications Conference*, Atlantic City, 1999.

[73] J. Chen, Y. Kuo, J. Li, F. Fu and Y. Ma, “Digital Modulation Identification by Wavelet Analysis,” in *Sixth International Conference on Computational Intelligence and Multimedia Applications*, Las Vegas, 2005.

[74] P. Prakasam and M. Madheswaran, “Automatic Modulation Identification of QPSK and GMSK using Wavelet Transform for Adaptive Demodulator in SDR,” in *International Conference on*

Signal Processing, Communications and Networking, Chennai, 2007.

[75] M. S. Muhlhaus, M. Oner, O. A. Dobre, H. U. Jkel and F. K. Jondral, "Automatic Modulation Classification for MIMO Systems Using Fourth-Order Cumulants," in *IEEE Vehicular Technology*, Quebec City, 2012.

[76] M. S. Mulhaus, M. Oner, O. A. Dobre, H. U. Jakel and F. K. Jondral, "A Novel Algorithm for MIMO Signal Classification using Higher-order Cumulants," in *IEEE Radio and Wireless Symposium*, Austin, 2013.

[77] D. Das, P. K. Bora and R. Bhattacharjee, "Cumulant Based Automatic Modulation Classification of QPSK, OQPSK, 8-PSK and 16-PSK," in *8th International Conference on Communication Systems and Networks*, Bangalore, 2016.

[78] H. C. Wu, M. Saquib and Z. Yun, "Novel Automatic Modulation Classification Using Cumulant Features for Communications via Multipath Channels," *IEEE Transactions on Wireless Communications,* vol. 7, no. 8, pp. 3098-3105, 2008.

[79] H. Ochiai, "High-order Statistical Analysis for Linearly Modulated Signals," in *Military Communications Conference*, Tampa, 2015.

[80] D. Digdarsini, M. Kumar, G. Khot, T. V. S. Ram and T. V. K, "FPGA Implementation of Automatic Modulation Recognition System for advanced SATCOM System," in *International Conference on Signal Processing and Integrated Networks*, Noida, 2014.

[81] J. Zhang and B. Li, "A New Modulation Identification Scheme for OFDM in Multipath Rayleigh Fading Channel," in *International Symposium on Computer Science and Computational Technology*, Shanghai, 2008.

[82] A. Swami and B. M. Sadler, "Hierarchical Digital Modulation Classification Using Cumulants," *IEEE Transactions on Communications,* vol. 48, no. 3, pp. 416-429, 2000.

[83] G. B. Markovic and M. L. Dukic, "Cooperative modulation classification with data fusion for multipath fading channels," *Electronics Letters,* vol. 49, no. 23, pp. 1494-1496, 2013.

[84] S. Amuru and C. R. C. M. da Silva, "Cumulant-based channel estimation algorithm for modulation classification in frequency-selective fading channels," in *Military Communication Conference (MILCOM 2012)*, Orlando, 2012.

[85] J. Venalainen, L. Terho and V. Koivunen, "Modulation classification in fading multipath channel," in *The Thirty-Sixth Asilomar Conference on Signals, Systems and Computers*, California, 2002.

[86] T. Hastie, R. Tibshirani and J. Friedman, "Model Assesment and Selection," in *The Elements of Statistical Learning: Data Mining, Inference and Prediction 2nd Ed*, New York, Springer, 2009, pp. 219-260.

[87] T. Hastie, R. Tibshirani and J. Friedman, “Overview of Supervised Learning,” in *The Elements of Statistical Learning: Data Mining, Inference and Prediction 2nd Ed.*, New York, Springer, 2009, pp. 9-42.

[88] C. M. Bishop, “Neural Networks,” in *Pattern Recognition and Machine Learning*, New York, Springer, 2006, pp. 225-290.

[89] S. J. Russel and P. Norvig, “Learning from Examples,” in *Artificial Intellegence: A Modern Approach 3rd Ed.*, Upper Saddle River, Prentice Hall, 2009, pp. 693-767.

[90] C. M. Bishop, “Introduction,” in *Pattern Recognition and Machine Learning*, New York, Springer, 2006, pp. 1-66.

[91] A. Hazza, M. Shoaib, S. A. Alshebeli and A. Fahad, “An Overview of Feature-based Methods for Digital Modulation Classification,” in *1st International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, Sharjah, 2013.

[92] S. Kharbech, I. Dayoub, M. Zwingelstein-Colin and E. P. Simon, “On Classifiers for Blind Feature-Based Automatic Modulation Classification over Multiple-Input–Multiple-Output Channels,” *IET Communications,* vol. 10, no. 7, pp. 790-795, 2016.

[93] P. V. Klaine, M. A. Imran, O. Onireti and R. D. Souza, “A Survey of Machine Learning Techniques Applied to Self Organizing Cellular Networks,” *IEEE Communications Surveys & Tutorials,* vol. (Still to be published), 2017.

[94] P. A. Harlianto, T. B. Adji and N. A. Setiawan, “Comparison of Machine Learning Algorithms for Soil Type Classification,” in *3rd International Conference on Science and Technology*, Yoqyakarta, 2017.

[95] T. Hastie, R. Tibshirani and J. Friedman, “Boosting and Additive Trees,” in *The Elements of Statistical Learning: Data Mining, Inference and Prediction 2nd Edition*, New York, Springer, 2009, pp. 337-388.

[96] C. M. Bishop, “Combining Models,” in *Pattern Recognition and Machine Learning*, New York, Springer, 2006, pp. 653-676.

[97] T. Hastie, R. Tibshirani and J. Friedman, “Additive Models, Trees and Related Methods,” in *The Elements of Statistical Learning: Data Mining, Inference and Prediction 2nd Ed.*, New York, Springer, 2009, pp. 295-336.

[98] C. M. Bishop, “Sparse Kernel Machines,” in *Pattern Recognition and Machine Learning*, New York, Springer, 2006, pp. 325-358.

[99] P. Kumbhar and M. Mali, “A Survey on Feature Selection Techniques and Classification Algorithms for Efficient Text Classification,” *International Journal of Science and Research,* vol. 5, no. 5, pp. 1267-1275, 2016.

[100] F. Benedetto, A. Tedeschi and G. Giunta, "Automatic Blind Modulation Recognition of Analog and Digital Signals in Cognitive Radios," in *84th Vehicular Technology Conference*, Montreal, 2016.

[101] L. De Vito, S. Rapuano and M. Villanacci, "Prototype of an Automatic Digital Modulation Classifier Embedded in a Real-Time Spectrum Analyzer," *IEEE Transactions on Instrumentation and Measurement,* vol. 59, no. 10, pp. 2639-2651, 2010.

[102] A. Hazza, M. Shoaib, A. Saleh and A. Fahd, "Classification of Digitally Modulated Signals in Presence of Non-Gaussian HF Noise," in *International Symposium on Wireless Communication Systems*, York, 2010.

[103] F. Huang, Z. Zhong, Y. Xu and G. Ren, "Modulation Recognition of Symbol Shaped Digital Signals," in *International Conference on Communications, Circuits and Systems*, Fujian, 2008.

[104] D. Boudreau, C. Dubuc, F. Patenaude, M. Dufour, J. Lodge and R. Inkol, "A Fast Automatic Modulation Recognition Algorithm and its Implementation in a Spectrum Monitoring Application," in *Military Communications Conference*, Los Angeles, 2000.

[105] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers and Electrical Engineering,* vol. 40, no. 1, pp. 16-28, 2014.

[106] R. O. Duda, P. E. Hart and D. G. Stork, "Nonmetric Methods," in *Pattern Classification*, New York, John Wiley & Sons, 2001, pp. 394-452.

[107] L. E. Frenzel, "Communication Receivers," in *Principles of Electronic Communication Systems*, New York, Mc Graw Hill, 2014, pp. 291-346.

[108] R. H. Hosking, Digital Receiver Handbook: Basics of Software Radio 6th Ed., New Jersey: Pentek, Inc., 2006.

[109] E. E. Azzouz and A. K. Nandi, "Mathematical Preliminaries," in *Automatic Modulation Recognition of Communication Signals*, Dordrecht, Kluwer Academic Publishers, 1996, pp. 9-12.

[110] A. K. Nandi and E. Azzouz, "Recognition of Digital Modulations," in *Automatic modulation recognition of communication signals*, Norwell, Kluwer Academic Publishers, 1996, pp. 77-107.

[111] Z. Zhu and A. K. Nandi, "Modulation Classification Features," in *Automatic Modulation CLassification: Principles, ALgorithms and Applications*, Chichester, John Wiley and SOns, 2015, pp. 65-80.

[112] Q. Mingwei, L. Xianglu and Y. Yuancheng, "An algorithm of digital modulation identification based on instantaneous features," *Journal of Theoretical and Applied Information Technology,* vol. 50, no. 2, pp. 396-400, 2013.

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

[114] "Fading Channels," Mathworks, 2017. [Online]. Available: http://www.mathworks.com/help/comm/ug/fading-channels.html#bq5zk36. [Accessed 16 October 2017].

[115] A. V. Oppenheim and R. W. Schafer, "Transform Analysis of Linear Time-Invariant Systems," in *Discrete-time Signal Processing Third Ed.*, Upper Saddle River, Pearson Education Inc, 2010, pp. 274-373.

[116] MATLAB, "R2017a," The MathWorks Inc., Natick, Massachusetts, 2017.

[117] MATLAB, "Statistics and Machine Learning Toolbox R2017a," The MathWorks Inc., Natick, Massachusetts, 2017.

[118] L. Breiman, J. Friedman and R. S. C. Olshen, Classification and Regression Trees, Boca Rotan: CRC Press, 1984.

[119] W. Y. Loh and Y. S. Shi, "Split Selection Methods for CLassification Trees," *Statistica Sinica,* vol. 7, pp. 815-840, 1997.

[120] W. Y. Loh, "Regression Trees with Unbiased Variable Selection and Interaction Detection," *Statistica Sinica,* vol. 12, pp. 361-386, 2002.

[121] X. Liu and J. W. W. Su, "A modulation recognition method based on carrier frequency estimation and decision theory," in *16th Asia-Pacific Conference on Communications*, Auckland, 2010.

[122] MATLAB, "Communications System Toolbox R2017a," The MathWorks Inc., Natick, Massachusetts, 2017.

[123] Xilinx, *Synthesis and Simulation Design Guide,* San Jose, California: Xilinx Inc., 2008.

[124] A. V. Oppenheim and R. W. Schafer, "Sampling of Continuous-Time Signals," in *Discrete-time Signal Processing Third Ed.*, Upper Saddle River, Pearson Education Inc., 2010, pp. 153-273.

[125] Xilinx, *7 Series DSP48E1 Slice User Guide,* San Jose, California: Xilinx Inc., 2016.

[126] A. K. Nandi and E. E. Azzouz, "Algorithms for Automatic Modulation Recognition of Communication Signals," *IEEE Transactions on Communications,* vol. 4, no. 46, pp. 431-436, 1998.

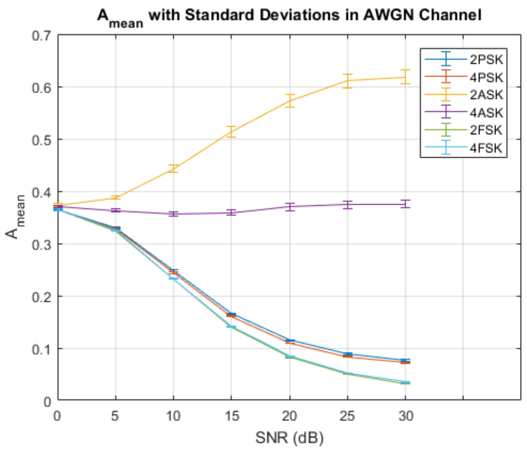
[127] A. K. Kumar, "SoC implementation of a Modulation Classification Module for Cognitive Radios," in *International Conference on Communication Systems and Networks (ComNet)*, Thiruvananthapuram, 2016.

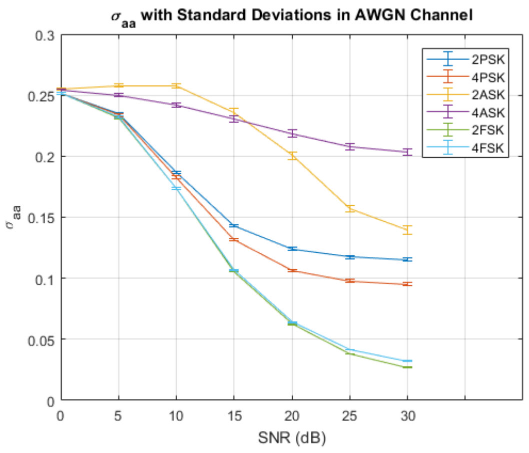
[128] L. Ladha and T. Deepa, “Feature Selection Methods and Algorithms,” *International Journal on Computer Science and Engineering*, vol. 3, no. 5, pp. 1787-1797, 2011.

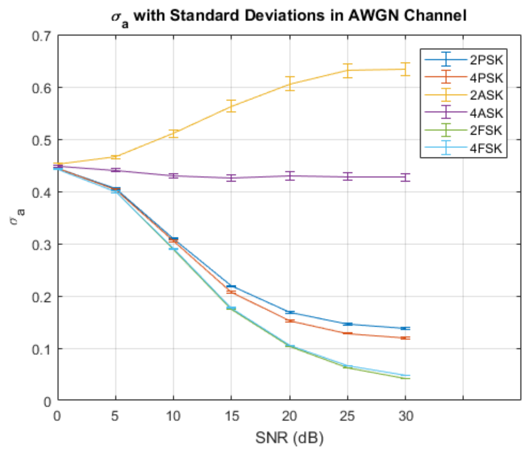
[129] M. W. Aslam, Z. Zhu and A. K. Nandi, “Automatic Digital Modulation Classification Using Genetic Programming with K-Nearest Neighbor,” in *Military Communications Conference*, San Jose, 2010.

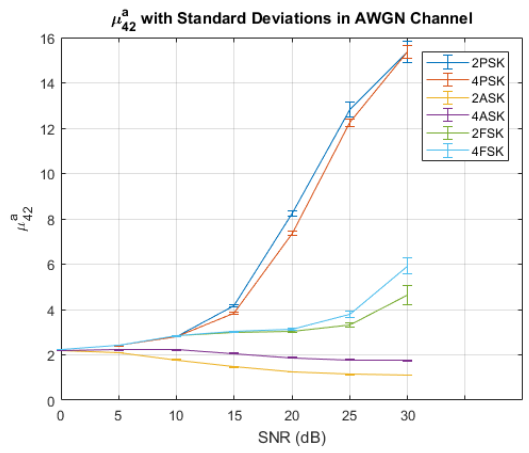
APPENDIX A

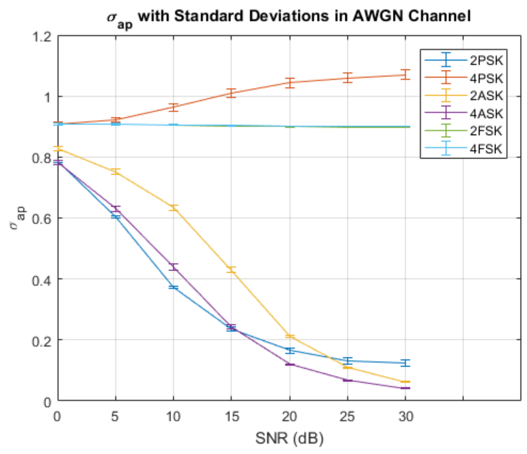
A.1.1 Matlab results with standard deviations in AWGN Channel

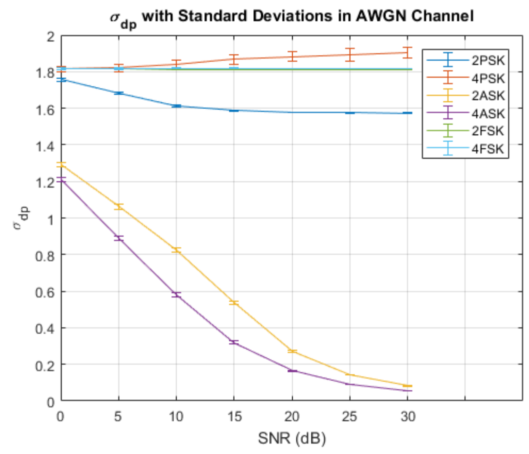


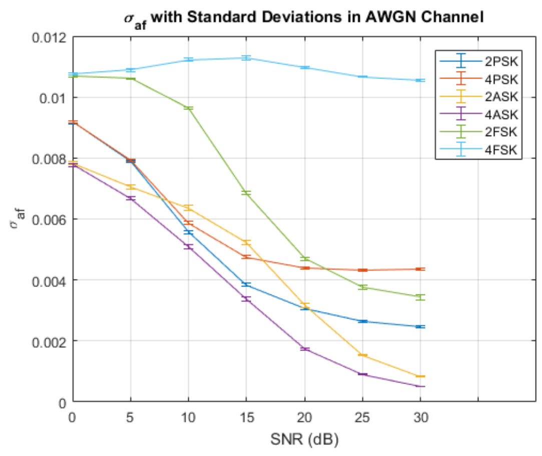


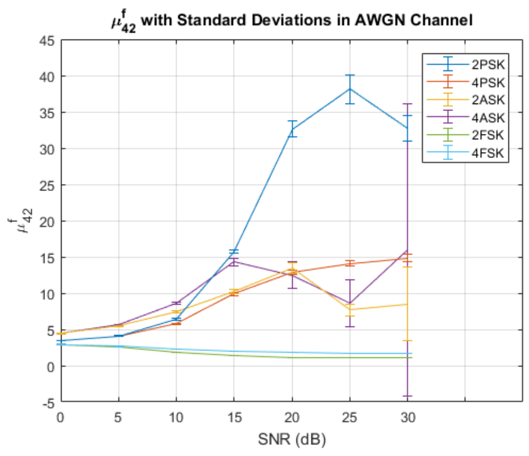




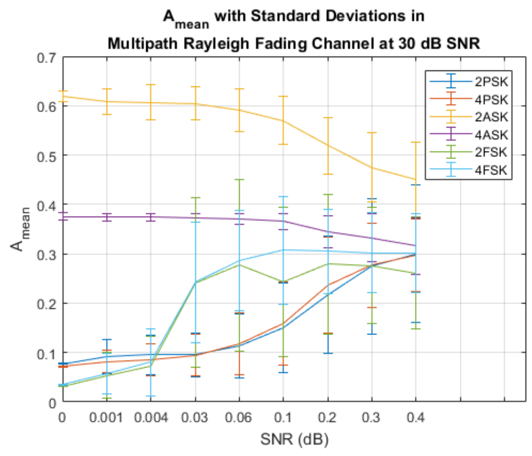


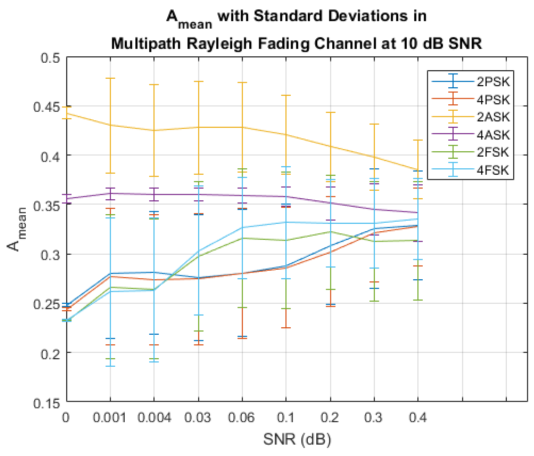


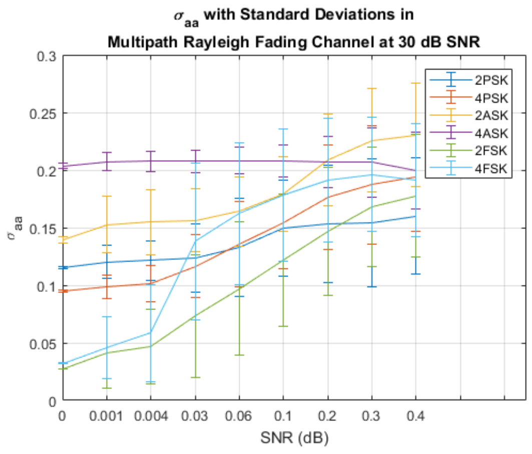


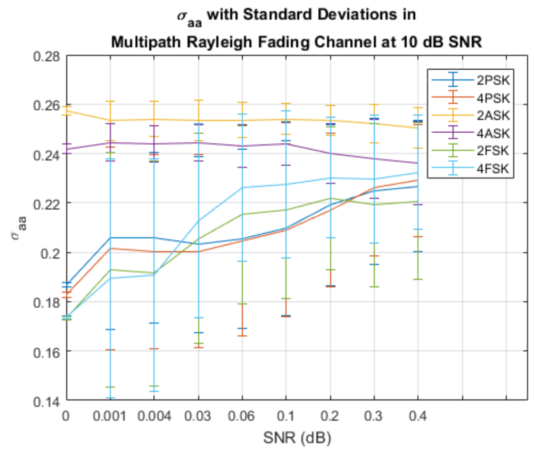


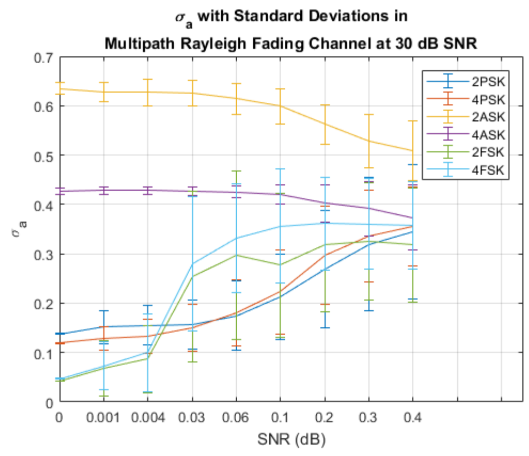
A.1.2 Matlab results with standard deviations in AWGN Channel

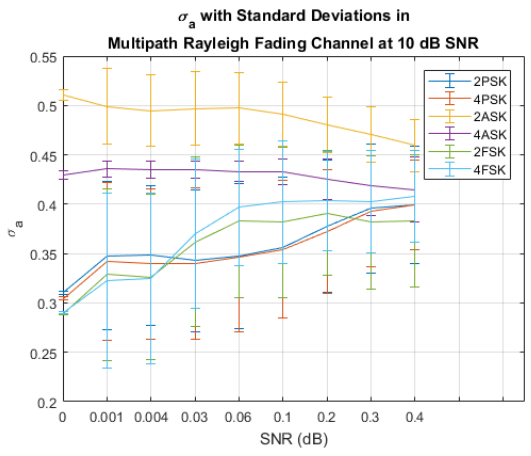


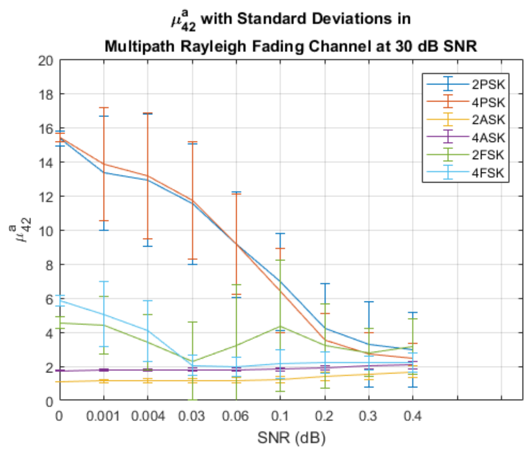


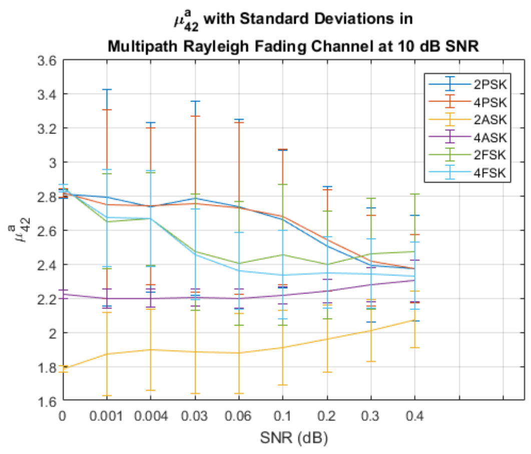


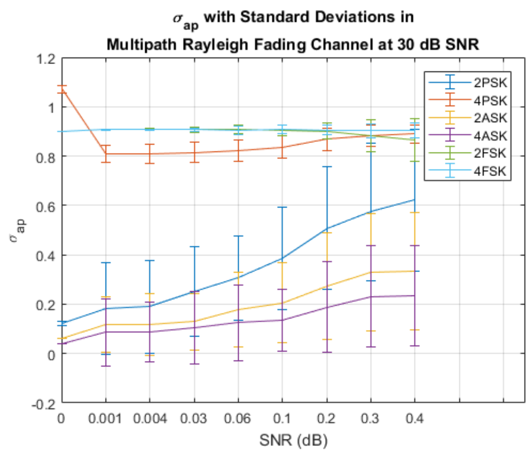


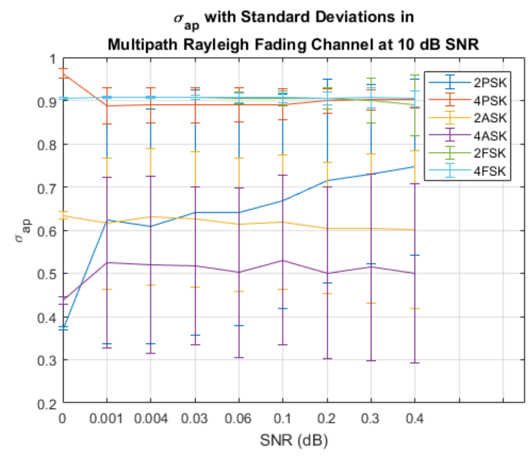

σ_a with Standard Deviations in
Multipath Rayleigh Fading Channel at 30 dB SNR
σ_a
SNR (dB)
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

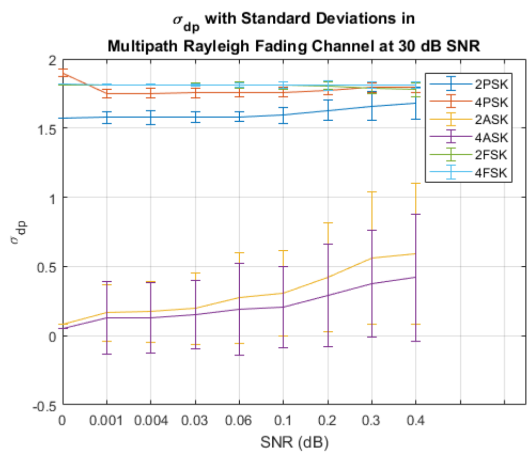

σ_a with Standard Deviations in
Multipath Rayleigh Fading Channel at 10 dB SNR
σ_a
SNR (dB)
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK


μ_{42}^{a} with Standard Deviations in
Multipath Rayleigh Fading Channel at 30 dB SNR
μ_{42}^{a}
SNR (dB)
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

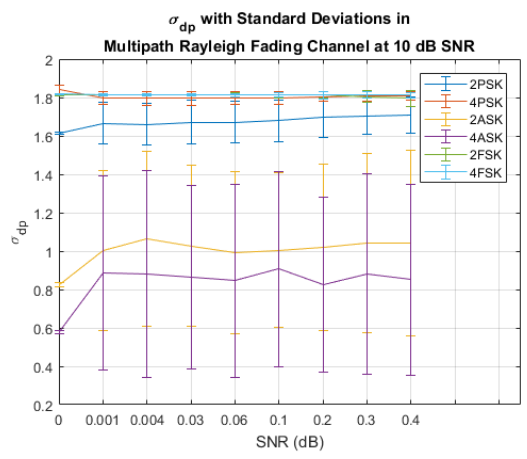

μ_{42}^{a} with Standard Deviations in
Multipath Rayleigh Fading Channel at 10 dB SNR
μ_{42}^{a}
SNR (dB)
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK


σ_{ap} with Standard Deviations in
Multipath Rayleigh Fading Channel at 30 dB SNR
σ_{ap}
SNR (dB)
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

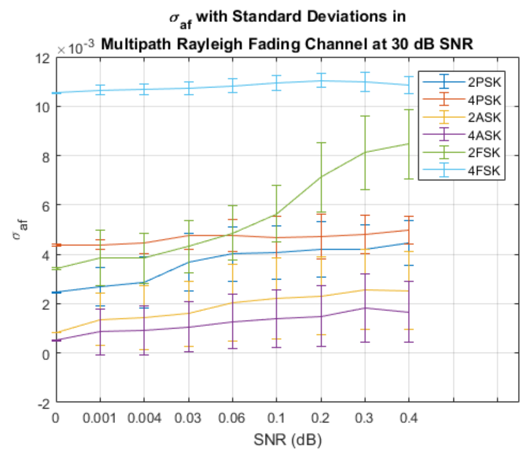

σ_{ap} with Standard Deviations in
Multipath Rayleigh Fading Channel at 10 dB SNR
σ_{ap}
SNR (dB)
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK



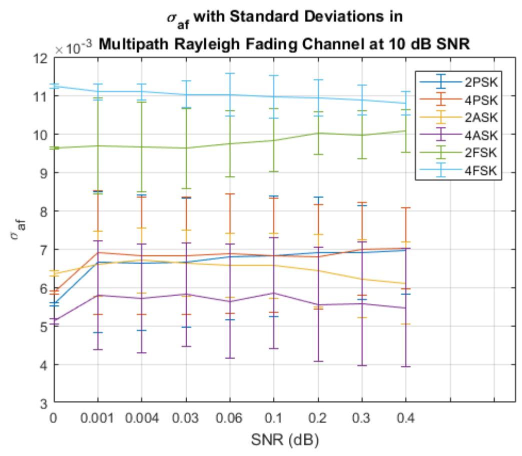
σ_{dp} with Standard Deviations in
Multipath Rayleigh Fading Channel at 30 dB SNR
σ_{dp}
SNR (dB)
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK



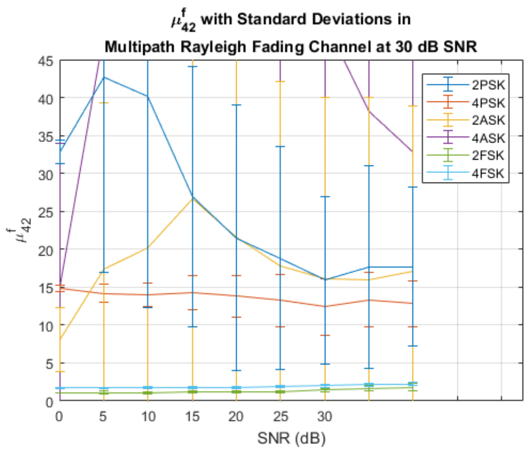
σ_{dp} with Standard Deviations in
Multipath Rayleigh Fading Channel at 10 dB SNR
σ_{dp}
SNR (dB)
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK



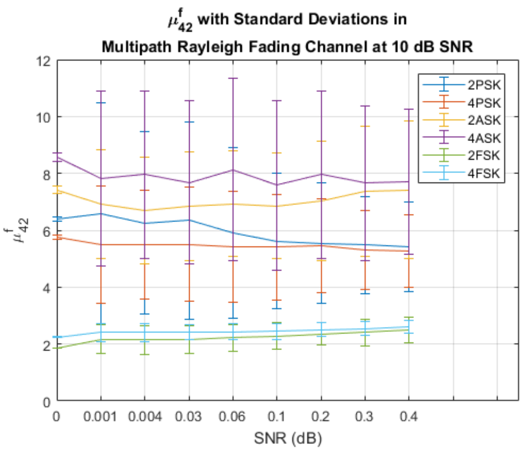
σ_{af} with Standard Deviations in
Multipath Rayleigh Fading Channel at 30 dB SNR
$\times 10^{-3}$
σ_{af}
SNR (dB)
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK



σ_{af} with Standard Deviations in
Multipath Rayleigh Fading Channel at 10 dB SNR
$\times 10^{-3}$
σ_{af}
SNR (dB)
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

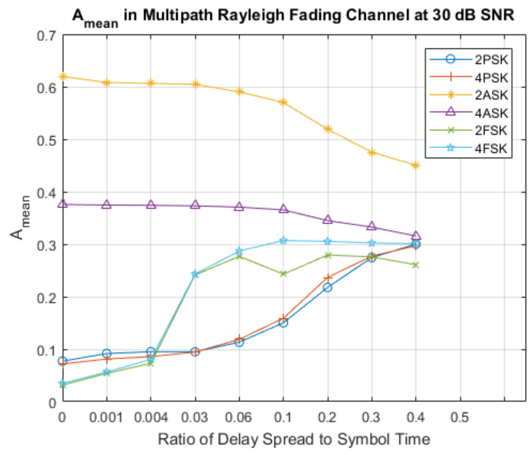


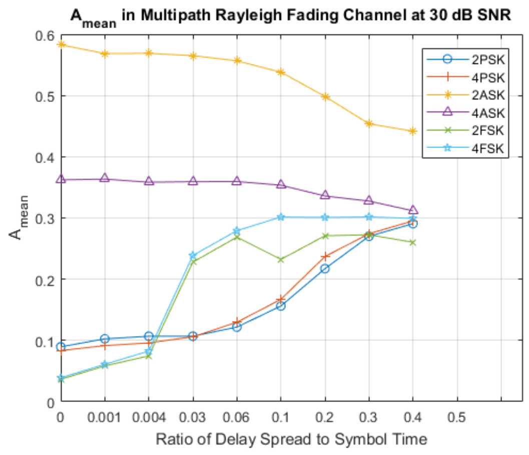
μ^{f}_{42} with Standard Deviations in
Multipath Rayleigh Fading Channel at 30 dB SNR
μ^{f}_{42}
SNR (dB)
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

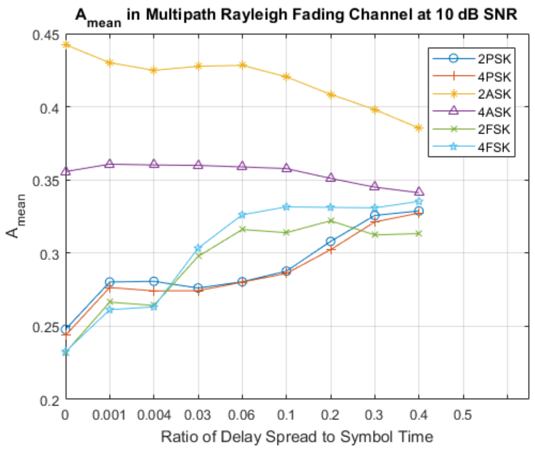


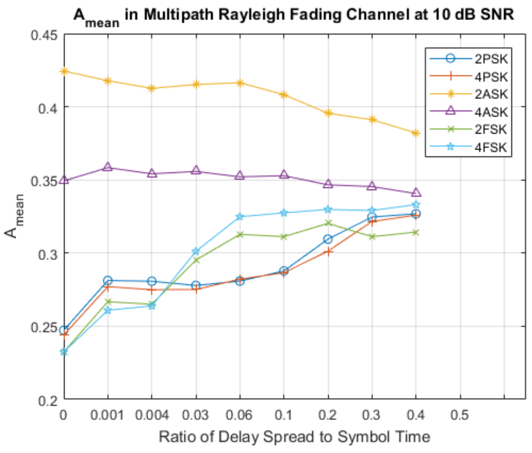
μ^{f}_{42} with Standard Deviations in
Multipath Rayleigh Fading Channel at 10 dB SNR
μ^{f}_{42}
SNR (dB)
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

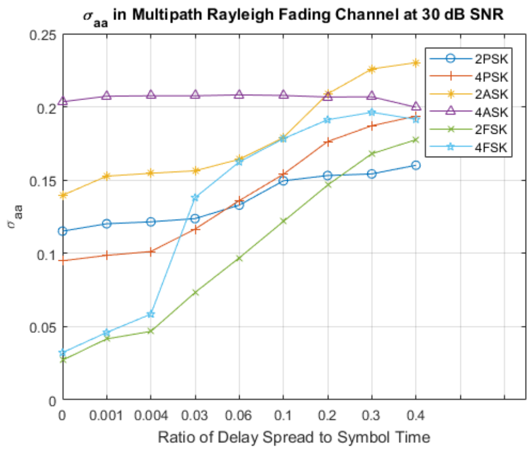
A.2. Matlab results (left) vs. Hardware results (right)

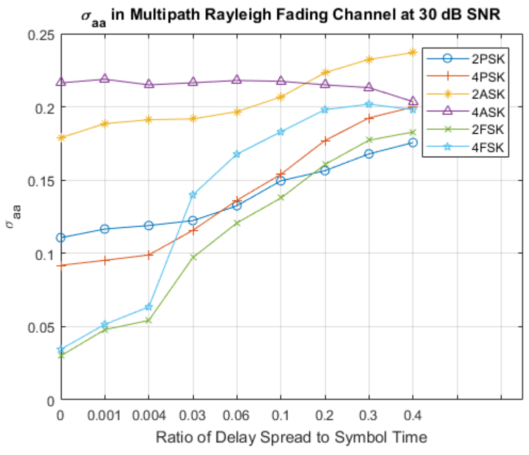


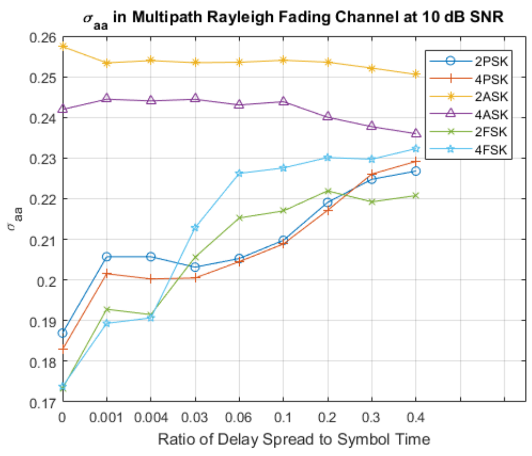


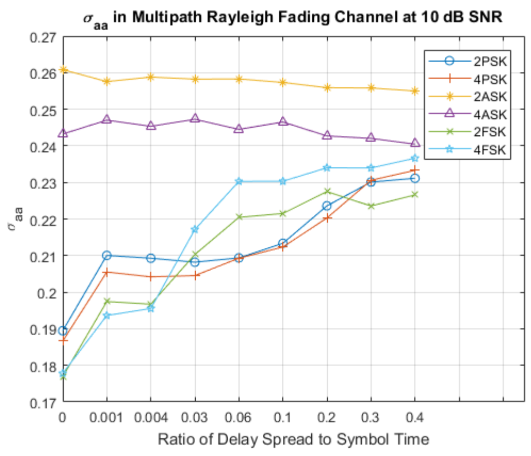


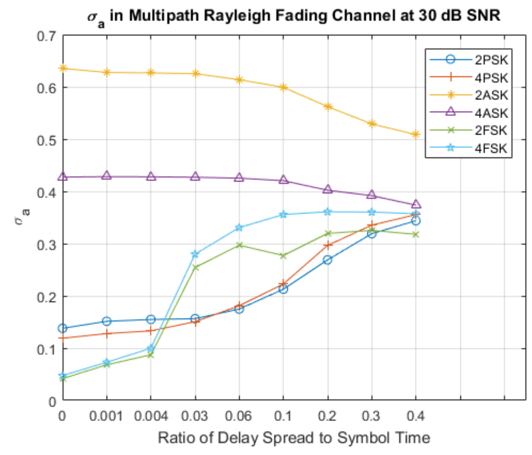


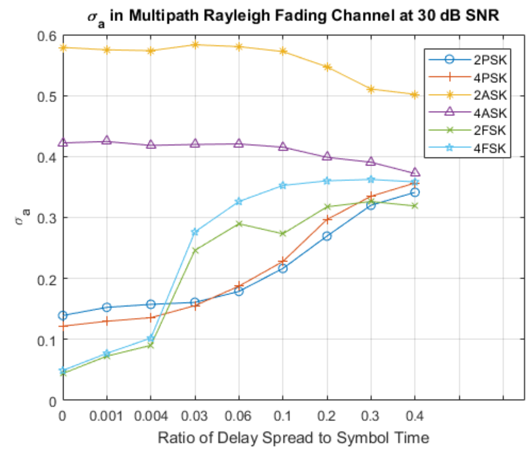


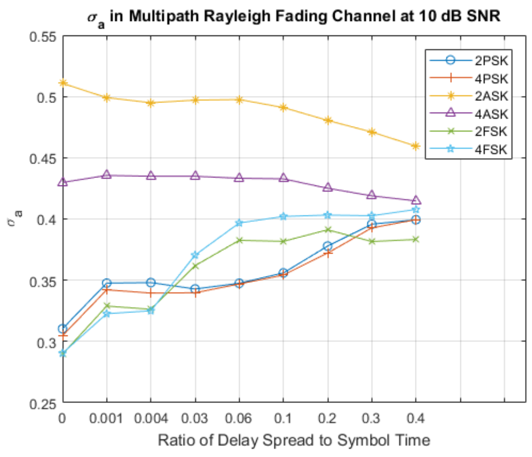


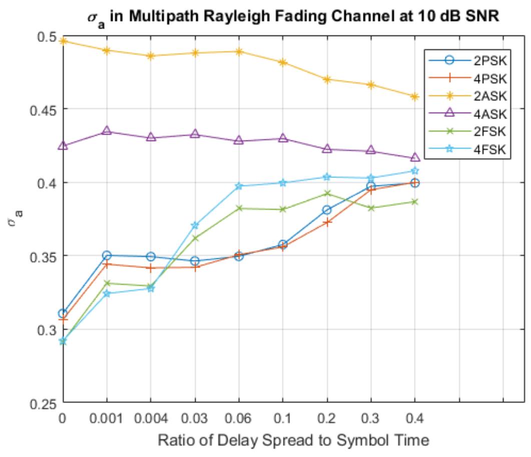

σ_{aa} in Multipath Rayleigh Fading Channel at 10 dB SNR
σ_{aa}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

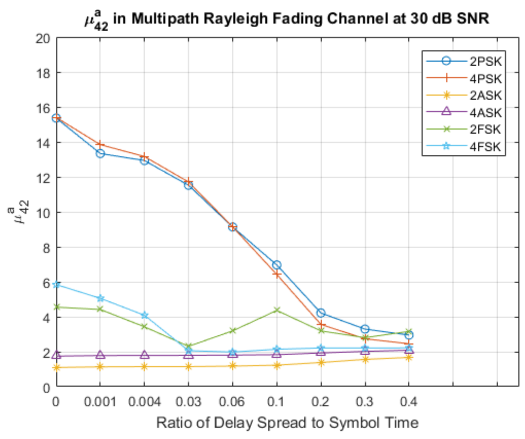

σ_{aa} in Multipath Rayleigh Fading Channel at 10 dB SNR
σ_{aa}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

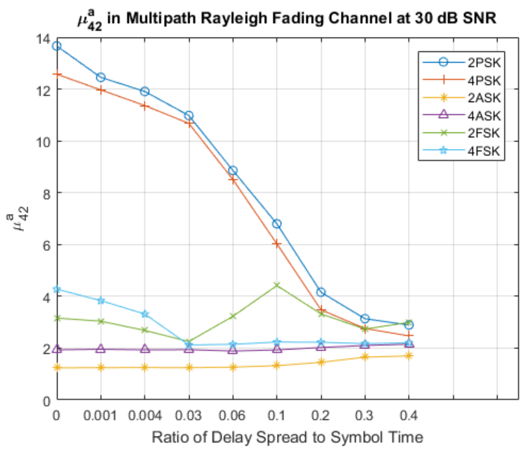

σ_a in Multipath Rayleigh Fading Channel at 30 dB SNR
σ_a
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

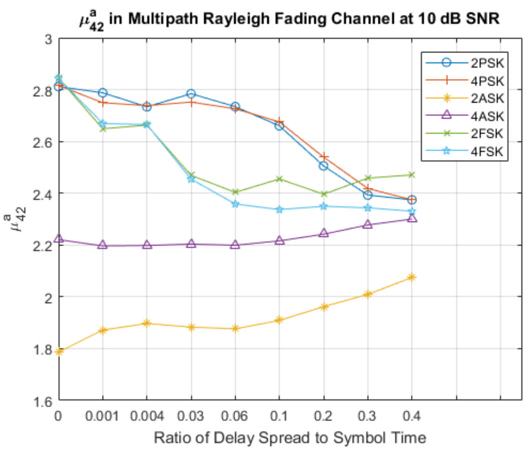

σ_a in Multipath Rayleigh Fading Channel at 30 dB SNR
σ_a
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

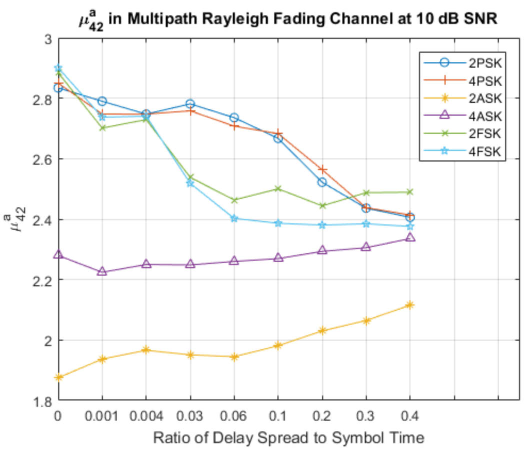

σ_a in Multipath Rayleigh Fading Channel at 10 dB SNR
σ_a
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

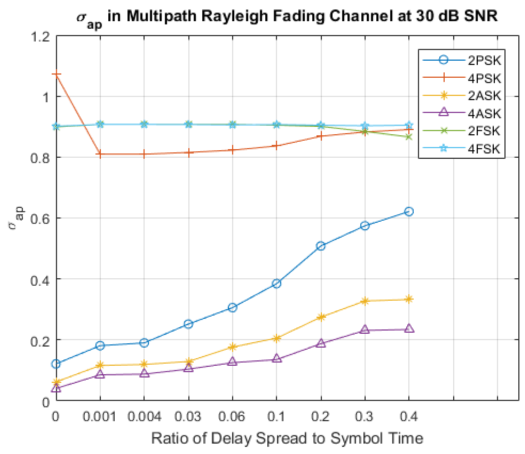

σ_a in Multipath Rayleigh Fading Channel at 10 dB SNR
σ_a
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

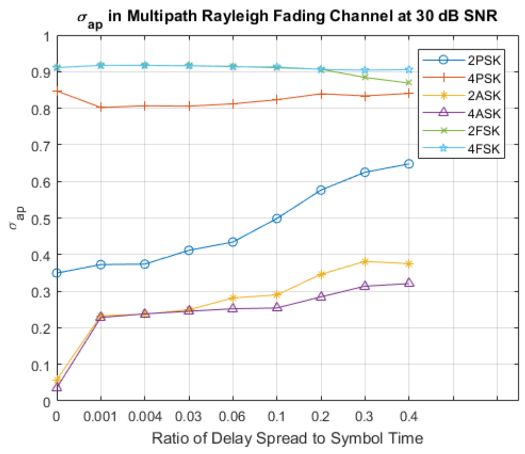

μ^a_{42} in Multipath Rayleigh Fading Channel at 30 dB SNR
μ^a_{42}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

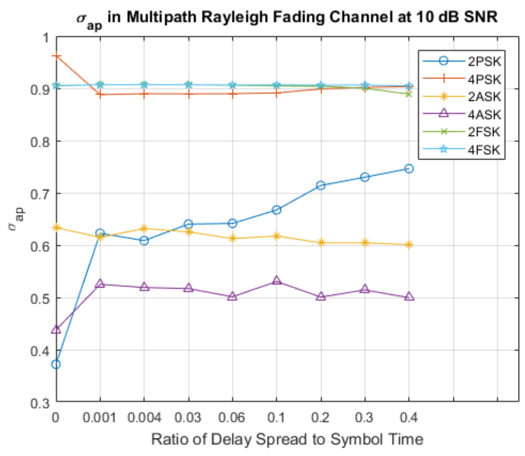

μ^a_{42} in Multipath Rayleigh Fading Channel at 30 dB SNR
μ^a_{42}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK


μ^a_{42} in Multipath Rayleigh Fading Channel at 10 dB SNR
μ^a_{42}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

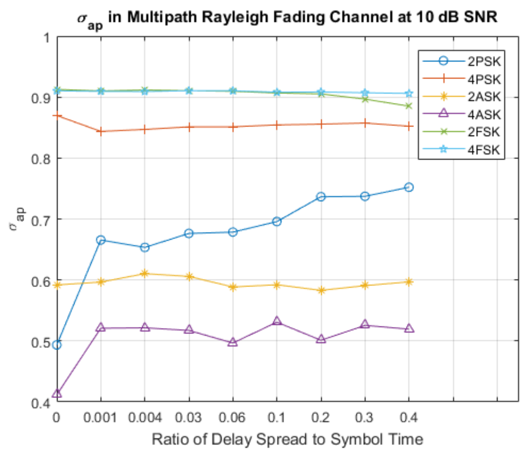

μ^a_{42} in Multipath Rayleigh Fading Channel at 10 dB SNR
μ^a_{42}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK


σ_{ap} in Multipath Rayleigh Fading Channel at 30 dB SNR
σ_{ap}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

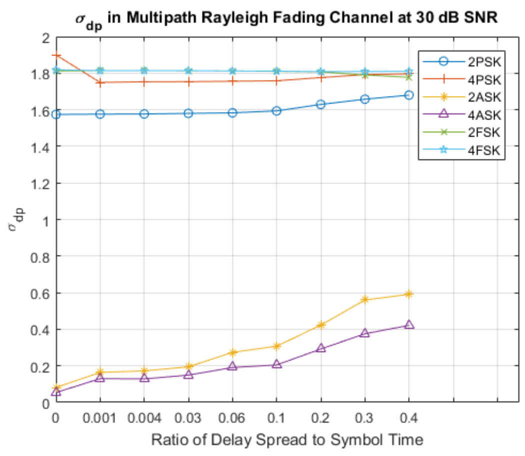

σ_{ap} in Multipath Rayleigh Fading Channel at 30 dB SNR
σ_{ap}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK



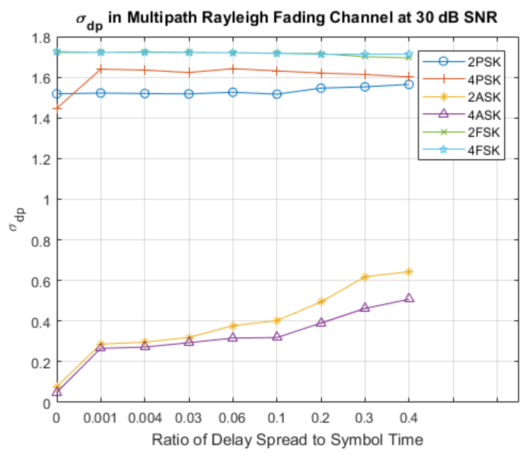
σ_{ap} in Multipath Rayleigh Fading Channel at 10 dB SNR
σ_{ap}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK



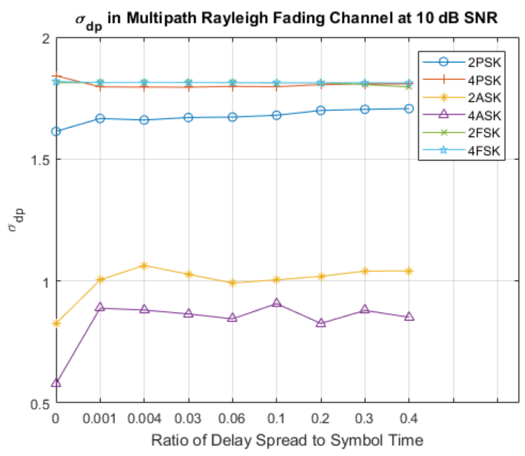
σ_{ap} in Multipath Rayleigh Fading Channel at 10 dB SNR
σ_{ap}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK



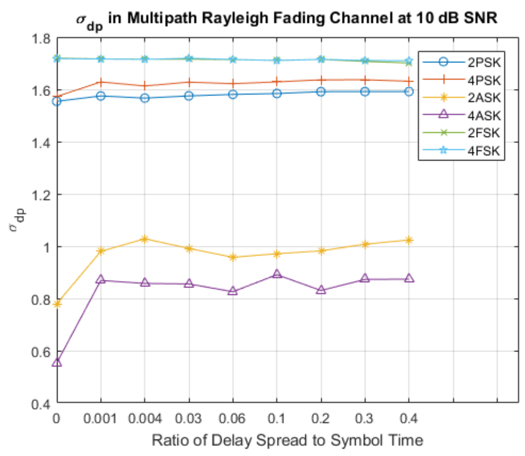
σ_{dp} in Multipath Rayleigh Fading Channel at 30 dB SNR
σ_{dp}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK



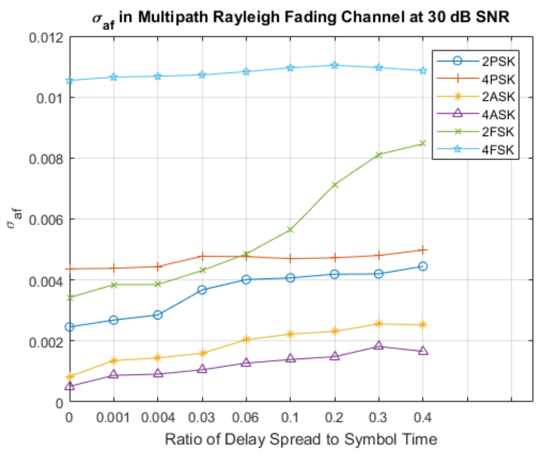
σ_{dp} in Multipath Rayleigh Fading Channel at 30 dB SNR
σ_{dp}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

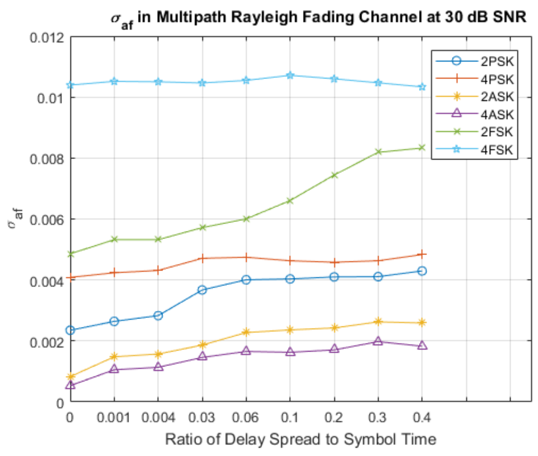


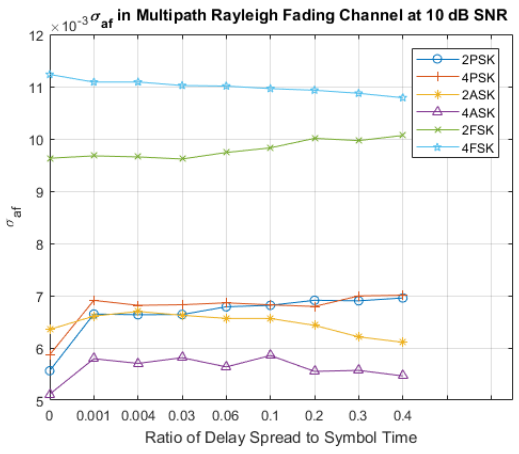
σ_{dp} in Multipath Rayleigh Fading Channel at 10 dB SNR
σ_{dp}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

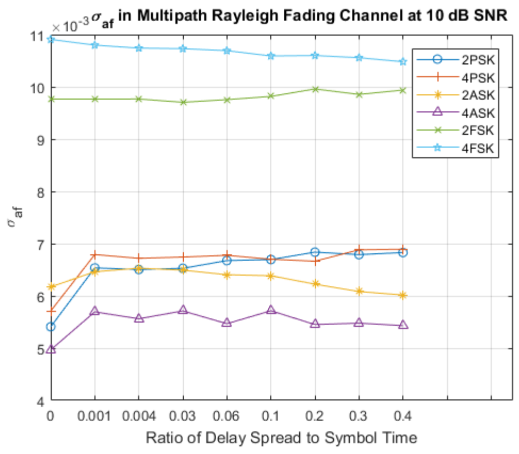


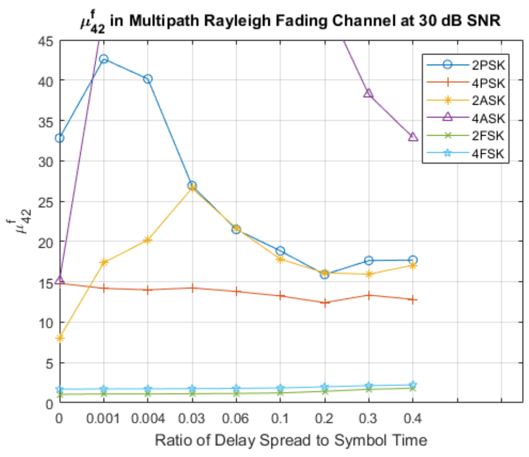
σ_{dp} in Multipath Rayleigh Fading Channel at 10 dB SNR
σ_{dp}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

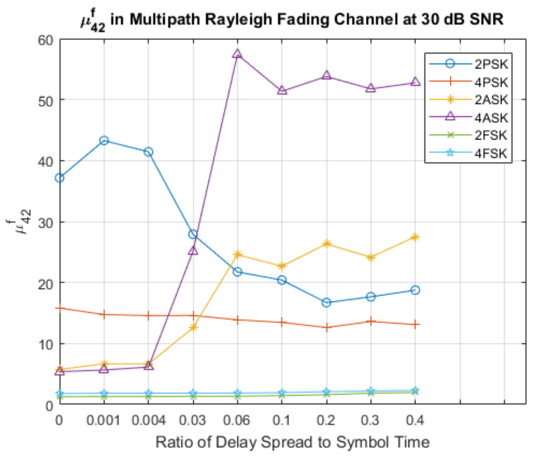

σ_{af} in Multipath Rayleigh Fading Channel at 30 dB SNR
σ_{af}
Ratio of Delay Spread to Symbol Time
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

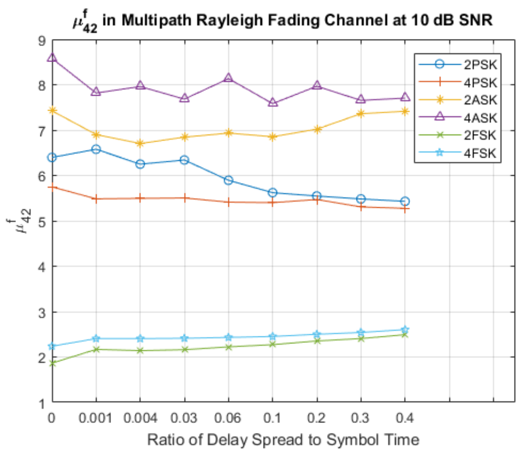

σ_{af} in Multipath Rayleigh Fading Channel at 30 dB SNR
σ_{af}
Ratio of Delay Spread to Symbol Time
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK


σ_{af} in Multipath Rayleigh Fading Channel at 10 dB SNR
$\times 10^{-3}$
σ_{af}
Ratio of Delay Spread to Symbol Time
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

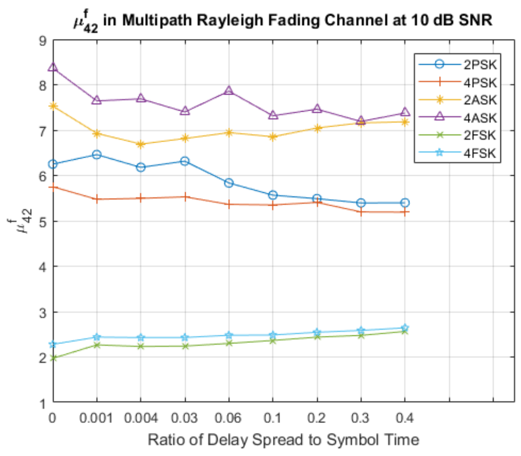

σ_{af} in Multipath Rayleigh Fading Channel at 10 dB SNR
$\times 10^{-3}$
σ_{af}
Ratio of Delay Spread to Symbol Time
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK


μ^f_{42} in Multipath Rayleigh Fading Channel at 30 dB SNR
μ^f_{42}
Ratio of Delay Spread to Symbol Time
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK


μ^f_{42} in Multipath Rayleigh Fading Channel at 30 dB SNR
μ^f_{42}
Ratio of Delay Spread to Symbol Time
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

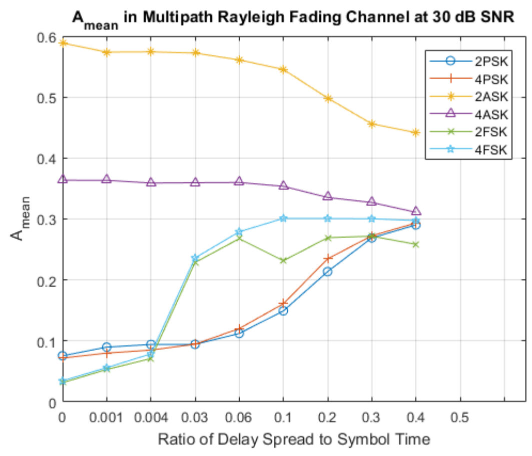


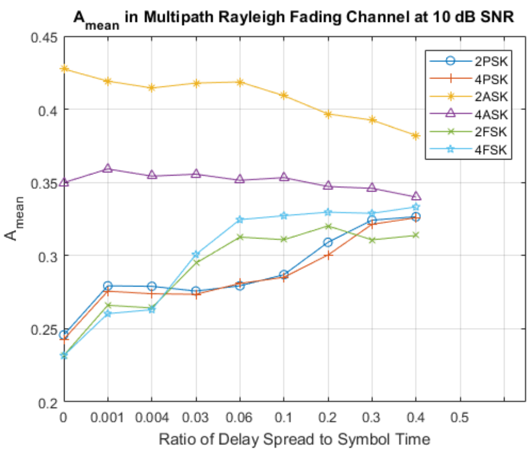
μ^f_{42} in Multipath Rayleigh Fading Channel at 10 dB SNR
μ^f_{42}
Ratio of Delay Spread to Symbol Time
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

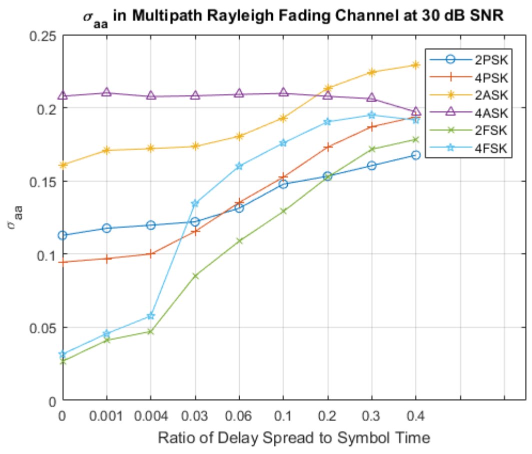


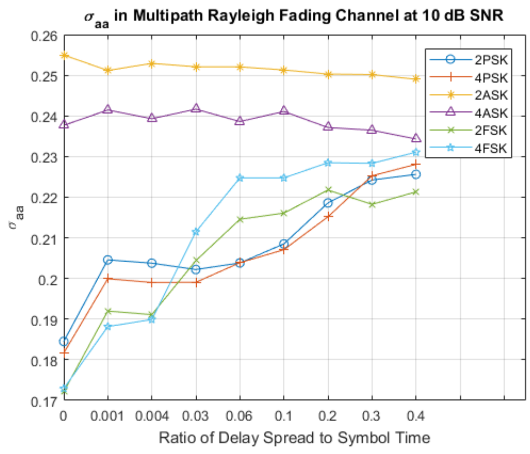
μ^f_{42} in Multipath Rayleigh Fading Channel at 10 dB SNR
μ^f_{42}
Ratio of Delay Spread to Symbol Time
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

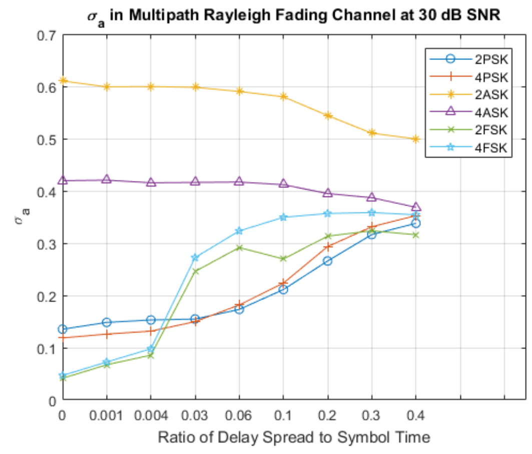
A.3. Results using 2048 samples in Matlab

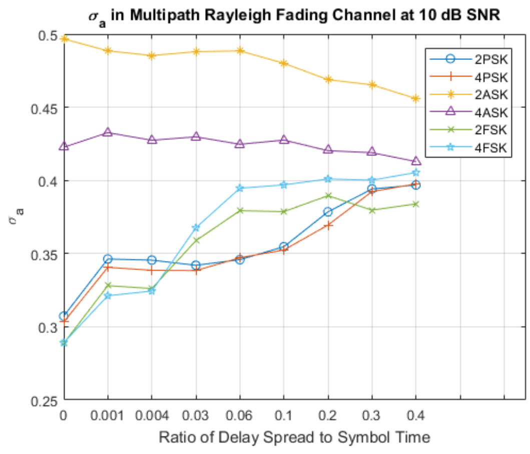


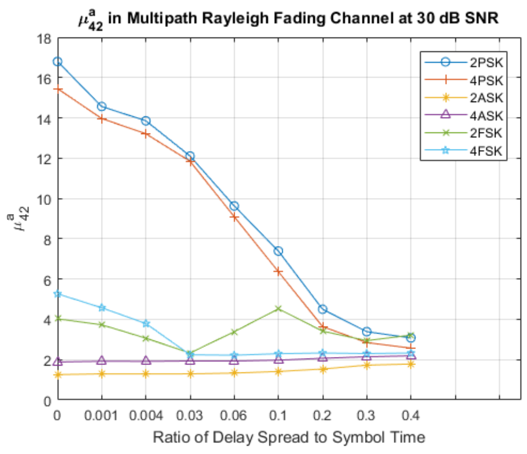


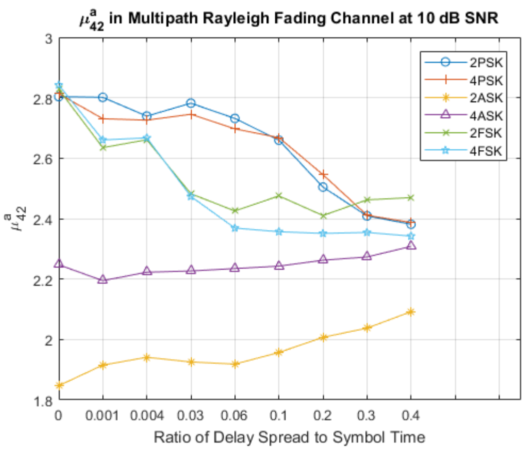


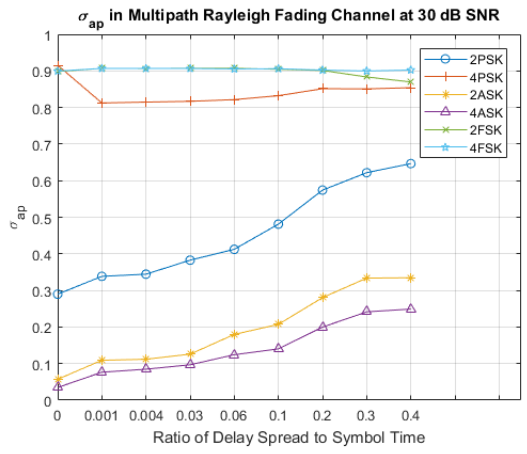


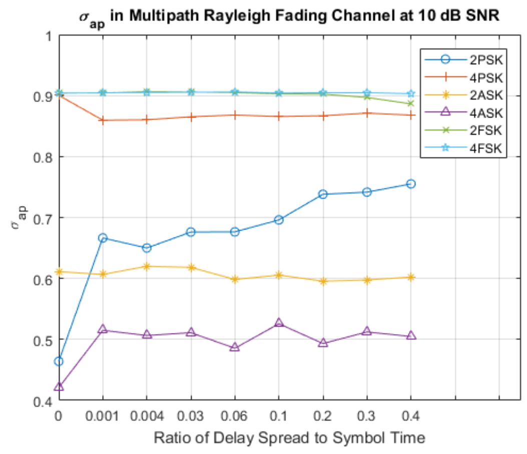


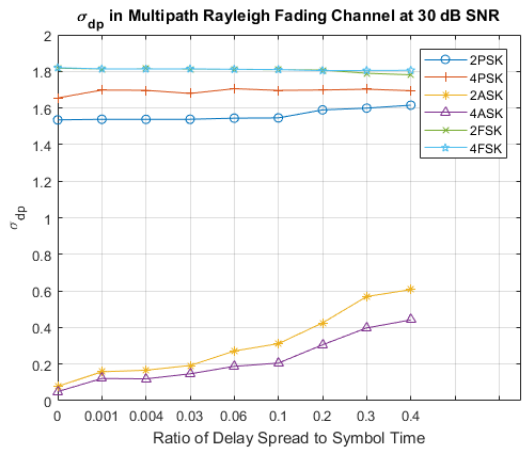


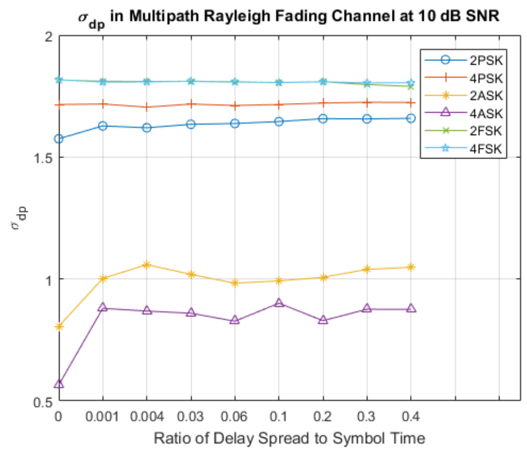

μ^a_{42} in Multipath Rayleigh Fading Channel at 30 dB SNR
μ^a_{42}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

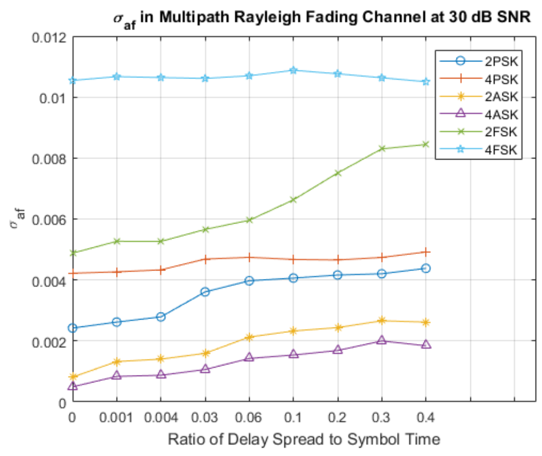

μ^a_{42} in Multipath Rayleigh Fading Channel at 10 dB SNR
μ^a_{42}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

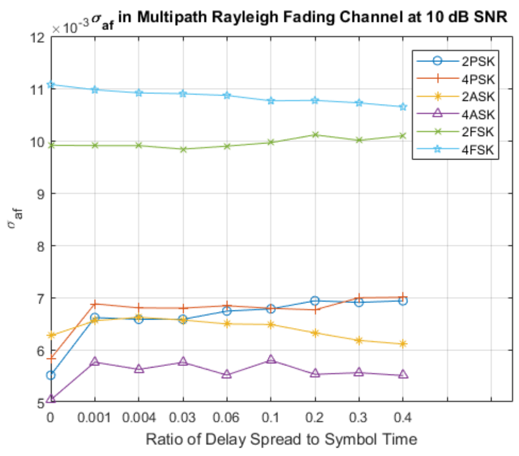

σ_{ap} in Multipath Rayleigh Fading Channel at 30 dB SNR
σ_{ap}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

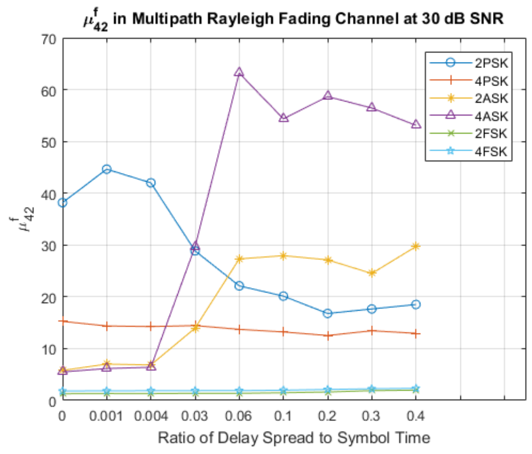

σ_{ap} in Multipath Rayleigh Fading Channel at 10 dB SNR
σ_{ap}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

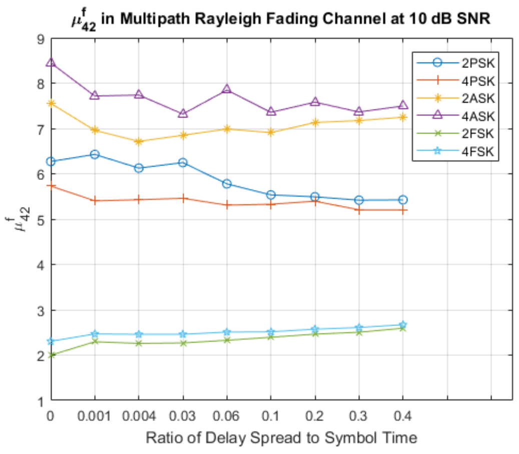

σ_{dp} in Multipath Rayleigh Fading Channel at 30 dB SNR
σ_{dp}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK


σ_{dp} in Multipath Rayleigh Fading Channel at 10 dB SNR
σ_{dp}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK


σ_{af} in Multipath Rayleigh Fading Channel at 30 dB SNR
σ_{af}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
0 0.002 0.004 0.006 0.008 0.01 0.012
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK


$\times 10^{-3}$ σ_{af} in Multipath Rayleigh Fading Channel at 10 dB SNR
σ_{af}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
5 6 7 8 9 10 11 12
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK


μ_{42}^{f} in Multipath Rayleigh Fading Channel at 30 dB SNR
μ_{42}^{f}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
0 10 20 30 40 50 60 70
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK


μ_{42}^{f} in Multipath Rayleigh Fading Channel at 10 dB SNR
μ_{42}^{f}
Ratio of Delay Spread to Symbol Time
0 0.001 0.004 0.03 0.06 0.1 0.2 0.3 0.4
1 2 3 4 5 6 7 8 9
2PSK
4PSK
2ASK
4ASK
2FSK
4FSK

A.4. The effect of noise filter bandwidth

Bandlimited noise with assumption (left) and bandlimited noise without assumption (right)

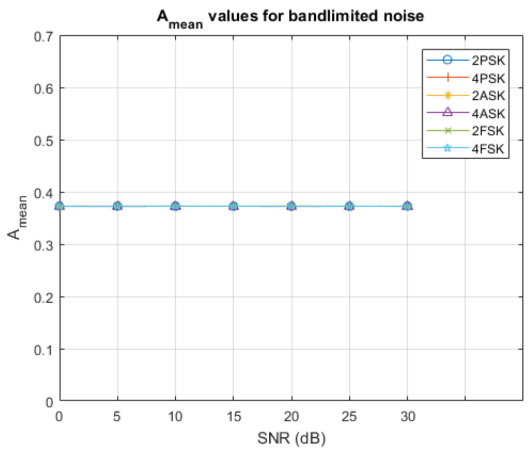


Figure 141: A_{mean} calculated for bandlimited noise with assumption

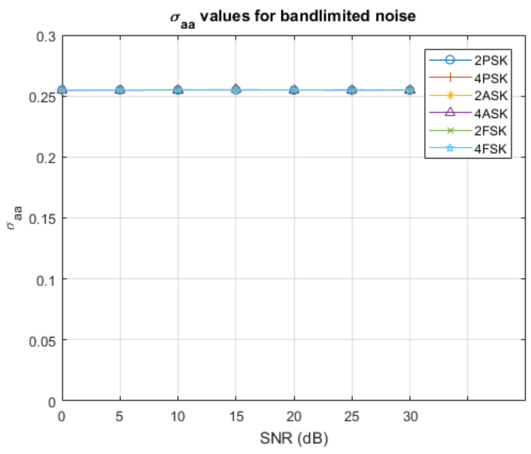


Figure 142: σ_{aa} calculated for bandlimited noise with assumption

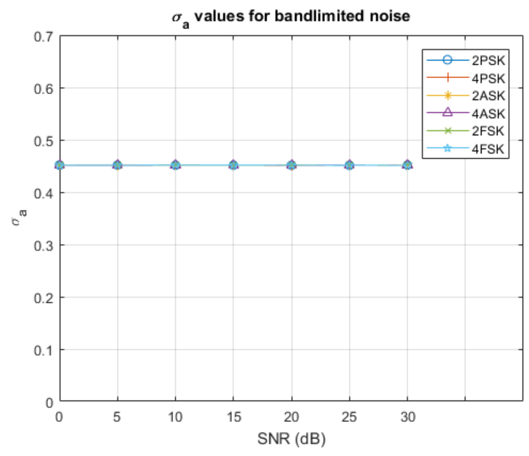


Figure 143: σ_a calculated for bandlimited noise with assumption

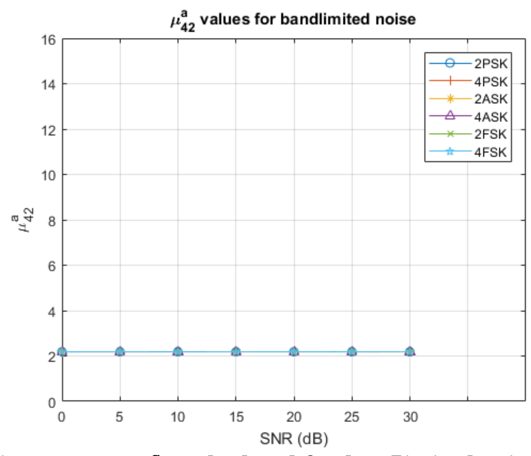


Figure 144: μ^a_{42} calculated for bandlimited noise with assumption

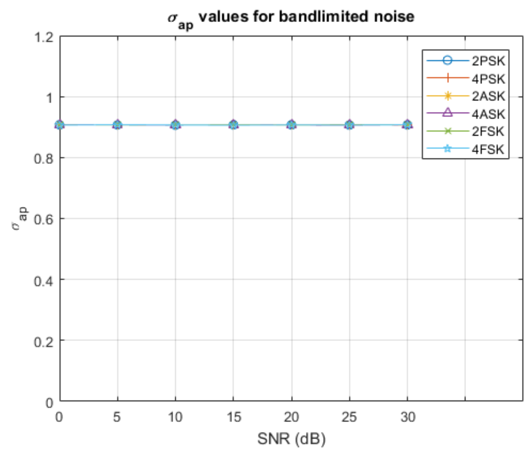


Figure 145: σ_{ap} calculated for bandlimited noise with assumption

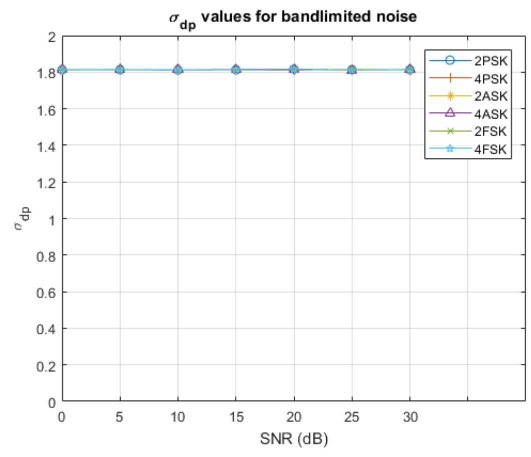


Figure 146: σ_{dp} calculated for bandlimited noise with assumption

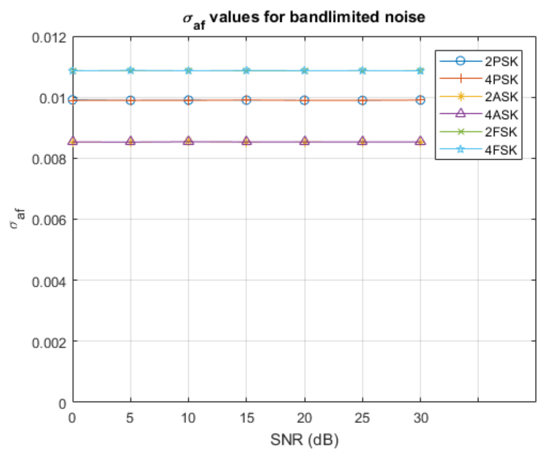


Figure 147: σ_{af} calculated for bandlimited noise with assumption

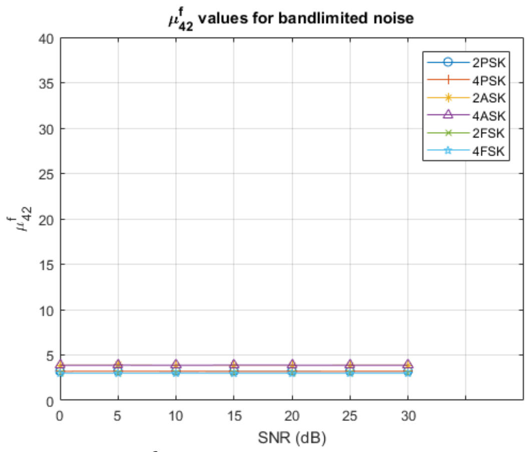


Figure 148: μ_{42}^{f} calculated for bandlimited noise with assumption

APPENDIX B

B.1. Confusion Matrices of Software Results

400 realisations (SNR):

0 dB	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	98	2	0	0	0	0
4ASK	2	98	0	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	0	0	100	0	0
2FSK	0	0	0	0	90	10
4FSK	0	0	0	0	9	91

Classification error = 3.83

20dB	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	0	100	0	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	0	0	100	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	0	100

Classification error = 0

5 dB	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	0	100	0	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	0	0	100	0	0
2FSK	0	0	0	0	98	2
4FSK	0	0	0	0	0	100

Classification error = 0.33%

25dB	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	0	100	0	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	0	0	100	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	0	100

Classification error = 0%

10 dB	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	0	100	0	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	0	0	100	0	0
2FSK	0	0	0	0	100	
4FSK	0	0	0	0	0	100

Classification error = 0%

30 dB	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	0	100	0	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	0	0	100	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	0	100

Classification error = 0%

15 dB	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	0	100	0	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	0	0	100	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	0	100

Classification error = 0%

400 realisations (fading at 30 dB SNR):

30 dB R_{DS}=0.00	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	0	100	0	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	0	0	100	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	0	100

Classification error = 0

30 dB R_{DS}=0.1	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	99	0	1	0	0	0
4ASK	2	98	0	0	0	0
2PSK	0	1	98	1	0	0
4PSK	0	0	1	99	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	1	99

Classification error = 1.17

30 dB R_{DS}=0.001	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	0	100	0	0	0	0
2PSK	0	0	99	1	0	0
4PSK	0	1	0	99	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	0	100

Classification error = 0.33

30 dB R_{DS}=0.2	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	6	91	2	1	0	0
2PSK	0	0	98	2	0	0
4PSK	0	0	1	99	0	0
2FSK	0	0	0	0	98	2
4FSK	0	0	0	0	2	98

Classification error = 2.67

30 dB R_{DS}=0.004	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	1	99	0	0	0	0
2PSK	0	0	99	1	0	0
4PSK	0	0	1	99	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	0	100

Classification error = 0.5

30 dB R_{DS}=0.3	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	86	11	2	1	0	0
4ASK	7	93	0	0	0	0
2PSK	3	4	86	7	0	0
4PSK	1	0	1	98	0	0
2FSK	0	0	0	0	96	4
4FSK	0	0	0	0	7	93

Classification error = 8.00

30 dB R_{DS}=0.03	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	0	100	0	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	0	0	98	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	1	99

Classification error = 0.5

30 dB R_{DS}=0.4	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	80	17	2	1	0	0
4ASK	7	91	2	0	0	0
2PSK	1	1	85	13	0	0
4PSK	0	0	2	98	0	0
2FSK	0	0	0	0	92	7
4FSK	0	0	0	0	10	90

Classification error = 10.67

30dB R_{DS}=0.06	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	2	98	0	0	0	0
2PSK	0	0	99	1	0	0
4PSK	0	0	1	99	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	1	99

Classification error = 0.83

400 realisations (fading at 10 dB SNR):

10 dB R_{DS}=0.00	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	100	0	0	0	0	0
4ASK	0	100	0	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	0	0	100	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	0	100

Classification error = 0%

10 dB R_{DS}=0.1	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	90	8	1	1	0	0
4ASK	7	89	3	1	0	0
2PSK	0	0	97	3	0	0
4PSK	0	0	1	99	0	0
2FSK	0	0	0	0	90	10
4FSK	0	0	0	0	15	85

Classification error = 8.33%

10 dB R_{DS}=0.001	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	91	8	1	0	0	0
4ASK	4	95	1	0	0	0
2PSK	0	0	97	3	0	0
4PSK	0	0	0	100	0	0
2FSK	0	0	0	0	91	9
4FSK	0	0	0	0	2	98

Classification error = 4.67%

10 dB R_{DS}=0.2	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	94	5	0	0	0	0
4ASK	14	85	1	0	0	0
2PSK	1	0	94	5	0	0
4PSK	1	0	0	99	0	0
2FSK	0	0	0	0	94	6
4FSK	0	0	0	0	17	83

Classification error = 8.5%

10 dB R_{DS}=0.004	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	96	4	0	0	0	0
4ASK	5	93	2	0	0	0
2PSK	0	0	99	1	0	0
4PSK	0	0	1	99	0	0
2FSK	0	0	0	0	97	3
4FSK	0	0	0	0	1	99

Classification error = 2.83%

10 dB R_{DS}=0.3	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	80	19	1	0	0	0
4ASK	18	79	3	0	0	0
2PSK	3	4	83	10	0	0
4PSK	2	1	3	94	0	0
2FSK	0	0	0	0	95	5
4FSK	0	0	0	0	14	86

Classification error = 13.83%

10dB R_{DS}=0.03	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	90	9	0	1	0	0
4ASK	3	97	0	0	0	0
2PSK	0	0	94	5	1	0
4PSK	0	0	1	100	0	0
2FSK	0	0	0	0	92	8
4FSK	0	0	0	0	6	94

Classification error = 5.5%

10 dB R_{DS}=0.4	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	62	36	2	0	0	0
4ASK	14	83	3	0	0	0
2PSK	4	4	82	10	0	0
4PSK	0	0	3	97	0	0
2FSK	0	0	1	1	96	2
4FSK	0	0	0	0	11	89

Classification error = 15.17%

10dB R_{DS}=0.06	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	97	3	0	0	0	0
4ASK	1	99	0	0	0	0
2PSK	0	0	99	1	0	0
4PSK	0	0	1	99	0	0
2FSK	0	0	0	0	92	8
4FSK	0	0	0	0	3	97

Classification error = 2.83%

B.2 Confusion Matrices of Hardware Results

400 realisations (fading at 30 dB SNR):

30 dB R_{DS}=0.00	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	89	0	0	0	0	0
4ASK	40	60	0	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	9	68	23	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	43	57

Classification error = 28.50

30 dB R_{DS}=0.001	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	88	11	0	0	0	0
4ASK	30	69	0	0	0	0
2PSK	0	3	96	1	0	0
4PSK	0	17	40	43	0	0
2FSK	0	0	1	0	97	2
4FSK	0	0	0	0	51	49

Classification error = 26.33

30 dB R_{DS}=0.004	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	90	8	0	0	2	0
4ASK	29	69	0	0	2	0
2PSK	0	5	95	1	0	0
4PSK	0	15	41	44	0	0
2FSK	0	0	2	1	95	2
4FSK	0	0	0	0	40	60

Classification error = 24.5

30 dB R_{DS}=0.03	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	95	5	0	0	0	0
4ASK	40	60	0	0	0	0
2PSK	0	1	99	0	0	0
4PSK	0	19	36	45	0	0
2FSK	0	0	1	0	97	2
4FSK	0	0	0	0	37	63

Classification error = 23.5

30dB R_{DS}=0.06	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	95	0	0	0	0	0
4ASK	28	72	0	0	0	0
2PSK	0	1	99	0	0	0
4PSK	0	17	29	54	0	0
2FSK	0	0	1	0	99	0
4FSK	0	0	0	0	38	62

Classification error = 19.83

30 dB R_{DS}=0.1	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	91	7	1	0	1	0
4ASK	22	77	0	0	1	0
2PSK	4	5	89	2	0	0
4PSK	0	8	42	50	0	0
2FSK	0	0	0	0	97	3
4FSK	0	0	1	0	23	76

Classification error = 20.00

30 dB R_{DS}=0.2	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	87	13	0	0	0	0
4ASK	14	86	2	1	0	0
2PSK	1	8	88	3	0	0
4PSK	0	14	51	35	0	0
2FSK	0	0	0	1	95	4
4FSK	0	0	0	1	37	62

Classification error = 24.50

30 dB R_{DS}=0.3	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	66	32	1	1	0	0
4ASK	16	84	0	0	0	0
2PSK	8	14	75	3	0	0
4PSK	4	20	49	27	0	0
2FSK	0	0	3	1	94	2
4FSK	0	0	2	2	33	63

Classification error = 31.83

30 dB R_{DS}=0.4	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	69	30	1	0	0	0
4ASK	10	89	1	0	0	0
2PSK	11	7	71	11	0	0
4PSK	6	18	49	27	0	0
2FSK	0	0	4	1	91	4
4FSK	0	0	2	1	34	63

Classification error = 31.67

400 realisations (fading at 10 dB SNR):

10 dB R_{DS}=0.00	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	91	9	0	0	0	0
4ASK	50	50	0	0	0	0
2PSK	0	8	92	0	0	0
4PSK	0	22	59	19	0	0
2FSK	0	0	0	0	88	12
4FSK	0	0	0	1	12	87

Classification error = 28.83%

10 dB R_{DS}=0.1	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	81	15	4	0	0	0
4ASK	38	54	8	0	0	0
2PSK	2	8	87	3	0	0
4PSK	5	23	47	25	0	0
2FSK	0	0	1	1	68	30
4FSK	0	0	1	0	28	71

Classification error = 35.67%

10 dB R_{DS}=0.001	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	83	10	6	1	0	0
4ASK	49	95	1	1	0	0
2PSK	0	13	83	3	1	0
4PSK	1	14	56	26	2	1
2FSK	0	0	0	0	73	27
4FSK	0	0	0	1	23	76

Classification error = 35.67%

10 dB R_{DS}=0.2	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	74	21	4	1	0	0
4ASK	47	46	7	0	0	0
2PSK	3	11	79	5	1	1
4PSK	4	16	52	28	0	0
2FSK	0	0	0	0	71	29
4FSK	0	0	0	0	29	71

Classification error = 38.5%

10 dB R_{DS}=0.004	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	81	14	4	1	0	0
4ASK	39	55	6	0	0	0
2PSK	0	12	86	2	0	0
4PSK	3	23	54	20	0	0
2FSK	0	0	1	0	70	29
4FSK	0	0	0	0	21	79

Classification error = 34.83%

10 dB R_{DS}=0.3	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	71	24	5	0	0	0
4ASK	46	51	3	0	0	0
2PSK	10	10	75	5	0	0
4PSK	5	19	57	19	0	0
2FSK	0	1	4	2	78	15
4FSK	0	1	3	1	32	63

Classification error = 40.50%

10dB R_{DS}=0.03	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	84	10	6	0	0	0
4ASK	50	46	4	0	0	0
2PSK	0	10	84	1	3	2
4PSK	1	13	59	22	3	2
2FSK	0	0	0	1	72	27
4FSK	0	0	0	0	21	79

Classification error = 35.5%

10 dB R_{DS}=0.4	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	68	30	2	0	0	0
4ASK	35	59	5	1	0	0
2PSK	17	11	69	3	0	0
4PSK	12	17	51	20	0	0
2FSK	0	0	2	1	77	20
4FSK	0	0	1	0	29	70

Classification error = 39.50%

10dB R_{DS}=0.06	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	87	12	1	0	0	0
4ASK	46	50	3	1	0	0
2PSK	0	11	86	2	0	1
4PSK	2	18	58	20	2	0
2FSK	0	0	3	1	72	24
4FSK	0	0	2	0	28	70

Classification error = 35.83%

B.3 Confusion Matrices of Hardware Results (2048 samples)

400 realisations (fading at 30 dB SNR):

30 dB R_{DS}=0.00	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	87	13	0	0	0	0
4ASK	1	98	1	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	0	8	92	0	0
2FSK	0	0	0	0	100	0
4FSK	0	0	0	0	1	99

Classification error = 4.00%

30 dB R_{DS}=0.001	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	88	10	2	0	0	0
4ASK	5	92	2	0	0	1
2PSK	0	0	98	2	0	0
4PSK	0	0	4	96	0	0
2FSK	0	0	0	0	98	2
4FSK	0	0	0	0	6	94

Classification error = 5.67%

30 dB R_{DS}=0.004	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	87	11	2	0	0	0
4ASK	16	81	1	1	0	1
2PSK	0	0	99	1	0	0
4PSK	0	0	0	100	0	0
2FSK	0	0	0	1	96	3
4FSK	0	0	0	0	4	96

Classification error = 6.83%

30 dB R_{DS}=0.03	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	96	4	0	0	0	0
4ASK	17	83	0	0	0	0
2PSK	0	0	96	4	0	0
4PSK	0	0	5	95	0	0
2FSK	0	0	0	0	99	1
4FSK	0	0	0	0	11	89

Classification error = 7.00%

30dB R_{DS}=0.06	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	93	3	1	0	0	0
4ASK	14	86	0	0	0	0
2PSK	0	0	98	2	0	0
4PSK	0	0	5	95	0	0
2FSK	0	0	0	0	99	1
4FSK	0	0	0	0	11	89

Classification error = 6.17%

30 dB R_{DS}=0.1	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	92	3	0	1	1	0
4ASK	16	82	2	0	0	0
2PSK	1	0	94	5	0	0
4PSK	0	0	3	97	0	0
2FSK	0	0	0	0	98	2
4FSK	0	0	0	1	9	90

Classification error = 7.83%

30 dB R_{DS}=0.2	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	86	12	2	0	0	0
4ASK	15	85	0	0	0	0
2PSK	2	1	78	19	0	0
4PSK	0	0	8	92	0	0
2FSK	0	0	0	1	92	7
4FSK	0	0	0	0	14	86

Classification error = 13.50%

30 dB R_{DS}=0.3	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	72	23	4	1	0	0
4ASK	15	85	0	0	0	0
2PSK	2	0	70	28	0	0
4PSK	6	0	4	90	0	0
2FSK	0	0	1	1	89	9
4FSK	0	0	1	2	19	78

Classification error = 19.33%

30 dB R_{DS}=0.4	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	64	31	4	1	0	0
4ASK	18	79	3	0	0	0
2PSK	3	1	57	39	0	0
4PSK	5	0	9	86	0	0
2FSK	0	0	3	1	88	8
4FSK	0	0	3	1	23	73

Classification error = 25.50%

400 realisations (fading at 10 dB SNR):

10 dB R_{DS}=0.00	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	91	9	0	0	0	0
4ASK	54	46	0	0	0	0
2PSK	0	0	100	0	0	0
4PSK	0	0	11	89	0	0
2FSK	0	0	0	0	87	13
4FSK	0	0	0	0	6	94

Classification error = 15.50%

10 dB R_{DS}=0.1	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	83	11	3	3	0	0
4ASK	39	48	8	5	0	0
2PSK	2	0	78	19	1	0
4PSK	7	0	8	85	0	0
2FSK	0	0	0	1	67	32
4FSK	0	0	3	0	27	70

Classification error = 28.17%

10 dB R_{DS}=0.001	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	82	12	2	4	0	0
4ASK	54	37	6	3	0	0
2PSK	4	0	80	14	2	0
4PSK	2	0	8	87	2	1
2FSK	0	0	0	0	67	33
4FSK	0	0	0	1	20	79

Classification error = 28.00%

10 dB R_{DS}=0.2	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	76	15	3	6	0	0
4ASK	42	50	5	3	0	0
2PSK	1	1	66	30	2	0
4PSK	2	0	10	88	0	0
2FSK	0	0	0	2	60	38
4FSK	0	0	0	1	23	76

Classification error = 30.67%

10 dB R_{DS}=0.004	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	85	12	1	2	0	0
4ASK	45	46	5	4	0	0
2PSK	2	0	79	19	0	0
4PSK	2	1	13	84	0	0
2FSK	0	0	0	0	66	34
4FSK	0	0	0	0	25	75

Classification error = 27.50%

10 dB R_{DS}=0.3	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	71	21	3	5	0	0
4ASK	44	50	2	4	0	0
2PSK	4	0	72	24	0	0
4PSK	1	1	16	82	0	0
2FSK	0	0	0	7	67	26
4FSK	0	0	4	2	26	68

Classification error = 31.67%

10dB R_{DS}=0.03	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	78	10	6	6	0	0
4ASK	50	44	5	1	0	0
2PSK	1	0	78	19	1	1
4PSK	1	1	10	86	2	0
2FSK	0	0	0	2	66	32
4FSK	0	0	0	0	27	73

Classification error = 29.17%

10 dB R_{DS}=0.4	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	67	27	2	4	0	0
4ASK	39	51	4	6	0	0
2PSK	16	0	53	31	0	0
4PSK	9	1	11	78	1	0
2FSK	0	0	4	2	75	19
4FSK	0	0	2	2	29	67

Classification error = 34.83%

10dB R_{DS}=0.06	2ASK	4ASK	2PSK	4PSK	2FSK	4FSK
2ASK	86	9	2	3	0	0
4ASK	47	45	5	3	0	0
2PSK	2	0	79	18	1	0
4PSK	1	0	12	86	1	0
2FSK	0	0	0	3	67	30
4FSK	0	0	2	1	21	76

Classification error = 26.83%