


Article

A Coherent Performance for Noncoherent Wireless Systems Using AdaBoost Technique

Heba Gamal ¹, Nour Eldin Ismail ², M. R. M. Rizk ², Mohamed E. Khedr ³
and Moustafa H. Aly ^{3,*} 

¹ Electrical Engineering Department, Faculty of Engineering, Pharos University, Alexandria, Egypt; hebagamal_17@hotmail.com

² Electrical Engineering, Department, Faculty of Engineering, Alexandria University, Alexandria, Egypt; noureldin.hassan@alexu.edu.eg (N.E.I.); mrm_rizk@mena.vt.edu (M.R.M.R.)

³ College of Engineering and Technology, Arab Academy for Science Technology and Maritime Transport, Alexandria, Egypt; khedr@aast.edu

* Correspondence: mosaly@aast.edu; Tel.: +20-100-663-9473

Received: 15 November 2018; Accepted: 6 January 2019; Published: 11 January 2019



Abstract: Boosting is a machine learning approach built upon the idea of producing a highly precise prediction rule by combining many relatively weak and imprecise rules. The Adaptive Boosting (AdaBoost) algorithm was the first practical boosting algorithm. It remains one of the most broadly used and studied, with applications in many fields. In this paper, the AdaBoost algorithm is utilized to improve the bit error rate (BER) of different modulation techniques. By feeding the noisy received signal into the AdaBoost algorithm, it is able to recover the transmitted data from the noisy signal. Consequently, it reconstructs the constellation diagram of the modulation technique. This is done by removing the noise that affects and changes the signal space of the data. As a result, AdaBoost shows an improvement in the BER of coherently detected binary phase shift keying (BPSK) and quadrature phase shift keying (QPSK). The AdaBoost is next used to improve the BER of the noncoherent detection of the used modulation techniques. The improvement appears in the form of better results of the noncoherent simulated BER in comparison to that of the theoretical noncoherent BER. Therefore, the AdaBoost algorithm is able to achieve a coherent performance for the noncoherent system. The AdaBoost is simulated for several techniques in additive white Gaussian noise (AWGN) and Rayleigh fading channels so, as to verify the improving effect of the AdaBoost algorithm.

Keywords: AdaBoost; BER; coherent; noncoherent; digital modulation techniques

1. Introduction

Boosting is a general technique used to increase the accuracy of any given learning algorithm. The Adaptive Boosting (AdaBoost) algorithm, introduced in 1995 by Freund and Schapire, has been used to solve many of the real-world problems of the earlier boosting algorithms. The AdaBoost algorithm was defined firstly for two class problems, but it can be continued for multi-class and regression problems [1,2]. When applied to two-class classification problems, AdaBoost has been shown to be extremely successful in creating accurate classifiers.

Freund and Schapire's AdaBoost algorithm for classification [3,4] has attracted much attention in the machine learning community [5] as well as in associated areas in statistics [6–8]. Different versions of the AdaBoost algorithm have proven to be very competitive in terms of the prediction accuracy in a diversity of applications.

Boosting approaches have been proposed as ensemble methods, which depend on the principle of generating multiple predictions and majority voting (averaging) among the individual classifiers.

Later, Breiman [6,7] made a path-breaking remark that the AdaBoost algorithm can be seen as a gradient descent algorithm in functional space, inspired by mathematical optimization and statistical estimation. Moreover, Friedman et al. [8] laid out more important foundations that related AdaBoost and other boosting algorithms to the framework of statistical estimation and additive basis expansion. In their terminology, boosting is characterized as “stagewise, additive modeling”. The word “additive” doesn’t indicate a model fit which is additive in the covariates, but points to the fact that boosting is an additive (in fact, a linear) combination of “simple” (function) estimators. Also, Ratliff et al. [9] and Wu et al. [10] established related ideas that were mostly acknowledged in the machine learning community. In [11], further views on boosting are given; specifically, the authors first pointed out the relation between boosting and L_1 -penalized estimation. The visions of Friedman et al. [8] opened new perceptions, namely to use boosting approaches in many contexts other than classification. We refer here to boosting methods for regression (including generalized regression) [12–14], for density estimation [15], for survival analysis [14,16], or for multivariate analysis [8,17]. Reference [18] presents a multidimensional back propagation neural network (BP-NN) based life calculation method that takes into consideration different driving currents and ambient temperatures. Since the well-known neural network (NN) can easily get trapped in local minima and be subjected to low precision, the BP-NN method is improved using the Adaboost algorithm. In some of these opinions, boosting is not only a black-box prediction tool, but it is also an estimation method for models with an explicit structure such as linearity or additivity. Boosting can then be considered as a very interesting regularization structure for estimating a model [12,16,19].

The modulation techniques occupy an important position in many applications. In [20], T. Liu et al. present a boosting algorithm as an ensemble frame to accomplish a higher accuracy than that of a single classifier. To calculate the effect of boosting algorithms, eight common communication signals are yet to be identified and five types of entropy are extracted as the training vector. Then, the AdaBoost algorithm based on a decision tree is used to confirm the idea of the boosting algorithm. The results show that AdaBoost is always a superior classifier.

In [21], a multiple-input multiple-output (MIMO) two-way relaying channel (TWRC) with physical layer network coding (PLNC) needs the recognition of a pair of source-modulations from the superposed constellation at the relay. The suggested algorithm is divided into two steps. The first stage uses the higher order statistics based features in conjunction with a genetic algorithm as a features-selection method, while the second one employs AdaBoost as a classifier. The obtained simulation results show the capability of the proposed algorithm to provide a decent recognition performance at acceptable signal-to-noise values.

Since the publishing of [21], the AdaBoost technique has been used in this line of work with noncoherent receivers for binary phase shift keying (BPSK) and quadrature phase shift keying (QPSK) modulation. It is important to include a brief survey about other noncoherent modulation schemes such as differential chaotic shift keying (DCSK), noise reduction DCSK (NR-DCSK), short reference-DCSK (SR-DCSK), permutation index DCSK (PI-DCSK) and multiuser orthogonal frequency division multiplexing DCSK (OFDM DCSK). The idea of considering the DCSK systems under pulse jamming environments was shown in [22], where the system was examined under slow- and fast-switching pulse jamming environments.

A new noncoherent scheme called PI-DCSK modulation was proposed in [23]. This inventive design scheme aims to improve data security, energy, and spectral efficiencies compared to its competitors. The NR-DCSK modulation was used in [24]. The antijamming performance of the NR-DCSK system was studied. Practical jamming environments were taken into consideration including broadband jamming (BBJ), partial-time jamming (PTJ), tone jamming (TJ), and sweep jamming (SWJ).

In [25], SR-DCSK was used to overcome the main drawbacks related to the low data rate and energy efficiency fondness of conventional DCSK systems. In [26], a multiuser OFDM-DCSK modulation was successfully established. It is a recent technique used to overcome the disadvantage of

the OFDM. The target of OFDM is to increase spectral and energy efficiencies by letting multiple access transmissions decrease the complexity using inverse fast Fourier transform/fast Fourier transform (IFFT/FFT) operations instead of parallel matched filters.

In this paper, the AdaBoost algorithm is used to improve the bit error rate (BER) of the different modulation techniques (BPSK and QPSK). For coherent detection, AdaBoost is able to improve the BER, and the simulated BER coincides with that of the theoretical BER curve of each technique. This is achieved by using the AdaBoost algorithm to recover the transmitted data from the noisy received signal. Consequently, it reconstructs the constellation diagram of the modulation technique by removing the noise that is affecting and changing the signal space of the data. This leads to an improvement in the BER of coherently detected BPSK and QPSK. Moving on to the next step, the AdaBoost is used to improve the BER of the noncoherent detection of the used modulation technique. The term noncoherent refers to detection of the received signal without the knowledge of the signal's phase. Instead of using known phase reference information at the receiver side, such as in the case of coherent detection, noncoherent detection uses the phase information of the prior symbol to detect the current one. As it is comparing two noisy signals with each other, consequently, the BER of noncoherent detection is estimated to be approximately two times worse than that of the coherent detection. Since the noncoherent system is worse in performance than the coherent one, when the AdaBoost algorithm is added, it improves the noncoherent detection's BER. This helps to improve the noncoherent detection of the receiver without adding more complexity, which was originally the advantage (i.e. lower complexity) of the noncoherent over the coherent detection.

The AdaBoost algorithm is able to achieve a coherent performance for the noncoherent system. The proposed system is simulated for different techniques in additive white Gaussian noise (AWGN) and Rayleigh fading channel to verify the improving effect of the AdaBoost algorithm. The novelty of using the AdaBoost algorithm is in detecting the signal's features by recovering the originally transmitted data from the noisy received signal and, therefore, helping to improve the BER. The innovation comes from using an algorithm that is well known in the field of image processing and feature recognition and applying it to wireless communication signals, where it can be a benefit from its boosting effect in improving the detected results.

The rest of this paper is organized as follows. In Section 2, the AdaBoost algorithm is explained. Digital modulation classification is then presented in Section 3. Section 4 discusses the simulation results. Section 5 is devoted to the main conclusion.

2. AdaBoost Algorithm

Boosting is a general term for a system performance enhancement. The principle of boosting is to effectively combine simple decision rules (so called weak classifiers) into strong ensembles of classifiers. This allows for the capturing of even quite complex decision boundaries. The Adaboost algorithm [27] (from adaptive boosting) is a powerful learning method with a built-in feature selection mechanism. The main concept of the AdaBoost learning algorithm is to create a strong classifier that has high detecting performance by joining weak classifiers. The AdaBoost algorithm learns by repeated calculations, and has two functions: choosing the feature and learning the classifier. Through reiteration of calculation, the simple classifying ability is reinforced, because weak classifiers that are an index of performance are added to the strong classifier in iteration [2]. After the initial step of learning, it updates a new weight to the previous classifier by using an emphasizing error. After learning, a strong classifier that forms the 2nd layer perceptron is acquired [28]. In the AdaBoost algorithm proved by Freund and Schapire, errors come close to zero when the number of iterations is increased exponentially [29].

Suppose we take a set of training data $(x_1, c_1), (x_2, c_2), \dots, (x_n, c_n)$, where the input (predictor variable) $x_i \in \mathbb{R}^p$, and the output (response variable) c_i is qualitative and assumes values in a finite set, e.g., $1, 2, \dots, K$ where K is the number of classes. The aim is to find a classification rule $C(x)$ from

the training data, so that when given a new input x , one can give it a class label c from $1, 2, \dots, K$ problems [1].

What is the best possible classification rule? A commonly used definition of the best classification rule is to achieve the bottommost misclassification error rate. Usually, it is supposed that the training data are independently and identically distributed samples from an unknown probability distribution $\text{Prob}(X,C)$. Then, the misclassification error rate for $C(x)$ [1] is:

$$E_{X,C} \mathbb{I}_{C(X) \neq C} = E_X \text{Prob}(C(X) \neq C|X) \tag{1}$$

$$E_{X,C} \mathbb{I}_{C(X) \neq C} = 1 - E_X \text{Prob}(C(X) = C|X) \tag{2}$$

$$E_{X,C} \mathbb{I}_{C(X) \neq C} = 1 - \sum_{k=1}^K E_X [\mathbb{I}_{C(X)=k} \text{Prob}(C = k|X)] \tag{3}$$

It is clear that $C^*(x) = \arg \max_k \text{Prob}(C = k|X = x)$ will reduce this quantity with the misclassification error rate equal to $1 - E_X \max_k \text{Prob}(C = k|X)$. This classifier is named the Bayes classifier, and the error rate it reaches is the Bayes error rate.

The AdaBoost algorithm is an iterative process that combines many weak classifiers to approximate the Bayes classifier $C^*(x)$. Beginning with the unweighted training sample, the AdaBoost builds a classifier, for instance, a classification tree, which produces class labels. If a training data point is misclassified, the weight of that training data point is enlarged (boosted). A second classifier is built using the new weights, which are no longer equal. Again, misclassified training data have their weights boosted and the process is repeated [1]. A description of the AdaBoost algorithm in the form of a flowchart is shown in Figure 1.

The following algorithm, Algorithm 1, shows the AdaBoost technique used to solve a classification problem of two classes. It is first used in case of two classes only and then it is developed to cover the multi-class problems as shown in Algorithm 2.

Algorithm 1: Two class AdaBoost [3]

- x_i : the input (predictor variable)
 - c_i : output (response variable)
 - w_i : observation weights
 - n : number of samples
 - M : number of iterations
 - $T(x)$: weak classifier
 - $err^{(m)}$: error rate
 - $\alpha(m)$: weight assigned to classifier
 - $C(x)$: final classification rule
 - K : number of classes ($1, 2, \dots, k$)
-

As shown in the steps of Algorithm 1, first the weight (w_i) of all samples is initialized, meaning that it is equal for all the samples.

1. Initialization of the observation weight $w_i = 1/n, i = 1, 2, \dots, n$.

Secondly, a number of iterations (M) are performed, where they go through all possible weak classifiers to find the best one that minimizes the error rate with respect to w_i .

2. For $m = 1$ to M :

(a) Fitting a classifier $T^{(m)}(x)$ to the training data using weights w_i .

(b) Calculate

$$err^{(m)} = \sum_{i=1}^n w_i \cdot \mathbb{1}(c_i \neq T^{(m)}(x_i)) / \sum_{i=1}^n w_i$$

The error rate $err^{(m)}$ is the summation of the previous process for all the samples.

(c) Calculate

$$\alpha^{(m)} = \log(1 - err^{(m)} / err^{(m)})$$

A weight $\alpha^{(m)}$ is assigned to the classifier.

(d) Set

$$w_i \leftarrow w_i \cdot \exp(\alpha^{(m)} \cdot \mathbb{1}(c_i \neq T^{(m)}(x_i))), i = 1, 2, \dots, n.$$

The weight w_i is then updated where the wrongly classified samples will have more weight.

(e) Re-normalization of w_i

3. Output

$$C(x) = \arg \max_k \sum_{m=1}^M \alpha^{(m)} \cdot \mathbb{1}(T^{(m)}(x) = k)$$

After M such iterations, the final classification rule $C(x)$ is formed.

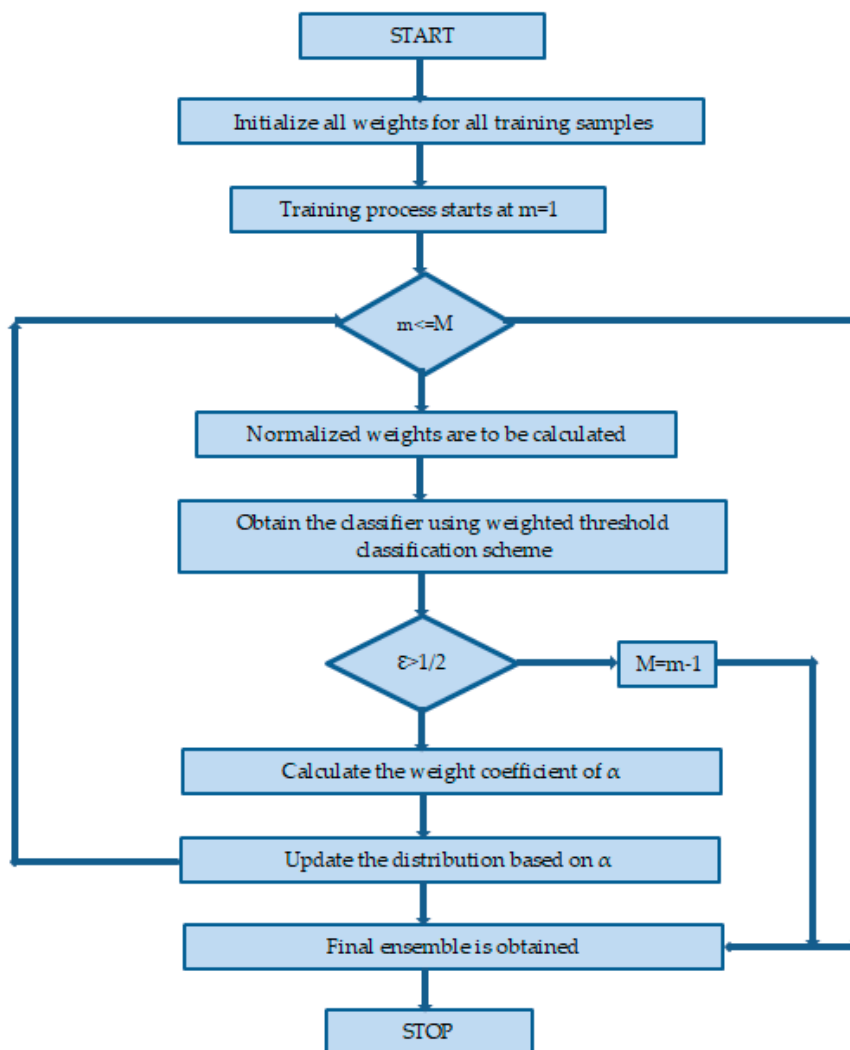


Figure 1. Adaptive Boosting (AdaBoost) algorithm flowchart.

Notice that in order for the misclassified training data to be boosted, it is necessary that the error of each weak classifier $err^{(m)}$ be less than $1/2$ (in reference to the distribution on which it was trained), otherwise $\alpha^{(m)}$ will be negative and the weights of the training samples will be updated in the wrong direction. For two-class classification problems, this condition is about the same as random guessing, but when $K > 2$, accuracy $1/2$ may be much harder to reach than the random guessing accuracy rate $1/K$. Therefore, AdaBoost may fail if the weak classifier $T(x)$ is not chosen correctly.

Typically, one may build 500 or 1000 classifiers in this way. A score is given to each classifier, and the final classifier is defined as the linear combination of the classifiers from every stage. Specifically, let $T(x)$ denote a weak multi-class classifier that gives a class label to x , then, the AdaBoost Algorithm 2 proceeds as follows [1]. It is a multi-class classifier algorithm, where the variable K is the number of classes. As for the overall sequence of the algorithm, it is almost the same as Algorithm 1.

Algorithm 2: Multi-Class AdaBoost [1]

- x_i : the input (predictor variable)
 - c_i : output (response variable)
 - w_i : observation weights
 - n : number of samples
 - M : number of iterations
 - $T(x)$: weak classifier
 - $err^{(m)}$: error rate
 - $\alpha^{(m)}$: weight assigned to classifier
 - $C(x)$: final classification rule
 - K : number of classes ($1, 2, \dots, k$)
-

As shown in the steps of Algorithm 1, first the weight (w_i) of all samples is initialized, meaning that it is equal for all the samples.

1. Initialization of the observation weight $w_i = 1/n, i = 1, 2, \dots, n$.

Secondly, a number of iterations (M) are performed, where they go through all possible weak classifiers to find the best one. This will minimize the error rate with respect to w_i .

2. For $m = 1$ to M :

- (a) Fitting a classifier $T^{(m)}(x)$ to the training data using weights w_i .
- (b) Calculate

$$err^{(m)} = \sum_{i=1}^n w_i \cdot \mathbb{1}(c_i \neq T^{(m)}(x_i)) / \sum_{i=1}^n w_i$$

The error rate $err^{(m)}$ is the summation of the previous process for all the number of samples.

- (c) Compute

$$\alpha^{(m)} = \log \frac{1 - err^{(m)}}{err^{(m)}} + \log(K - 1)$$

A weight $\alpha^{(m)}$ is assigned to the classifier, the term $\log(K - 1)$ is added to consider the number of classes in this algorithm as it is a multiclass algorithm.

- (d) Set

$$w_i \leftarrow w_i \cdot \exp\left(\alpha^{(m)} \cdot \mathbb{1}(c_i \neq T^{(m)}(x_i))\right), i = 1, \dots, n$$

The weight w_i is then updated where the wrongly classified samples will have more weight.

- (e) Re-normalization of w_i

3. Output

$$C(x) = \arg \max_k \sum_{m=1}^M \alpha^{(m)} \cdot \mathbb{1}_{\left(T^{(m)}(x) = k\right)}$$

After M such iterations, the final classification rule $C(x)$ is formed.

3. Digital Modulation Classification

Automatically recognizing the modulation technique of a detected signal, the midway step between signal detection and demodulation, is the main task of an intelligent receiver. Clearly, with no information of the transmitted data and many unknown parameters at the receiver, such as the signal power, carrier frequency, phase offsets, and timing information, etc., a blind identification of the modulation format is a challenging task. This becomes even more challenging in real-world situation [30].

Recognition of the modulation type of an unknown signal offers valuable insight into its structure, source, and properties. If the modulation type of an intercepted signal is extracted, jamming can be carried out more accurately by focusing all resources into essential signal parameters.

Modulation is the process of varying a periodic waveform, i.e., a tone, in order to use that signal to convey a message. The most essential digital modulation techniques are Amplitude Shift Keying (ASK), Frequency Shift Keying (FSK), Phase Shift Keying (PSK), and Quadrature-Amplitude Modulation (QAM). Modulation recognition is an intermediate step on the path to complete message recovery. Therefore, correct retrieval of the message per se is not an objective or even a requirement [31,32].

Early on it was known that modulation classification is, first and foremost, a classification problem well-suited to pattern recognition algorithms. A successful statistical classification needs the correct set of features extracted from the unidentified signal. This is where the use of classification algorithms such as AdaBoost arises. The proposed model in this paper is shown in Figure 2.

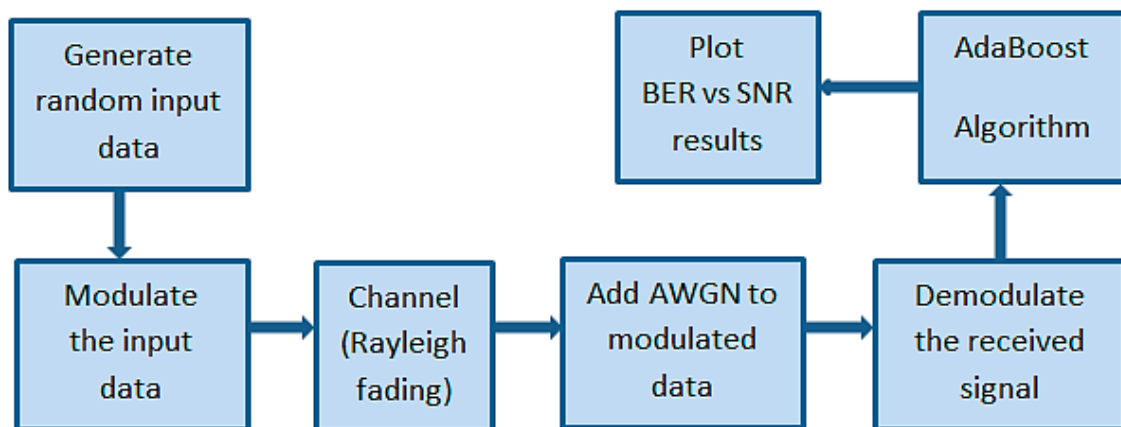


Figure 2. Flowchart of the proposed system. AWGN = additive white Gaussian noise; BER = bit error rate; SNR = signal-to-noise ratio.

The system is built up as follows:

1. Generate random input data.
2. Modulate the input data using BPSK or QPSK (coherent and noncoherent).
3. Add AWGN to the modulated data.
4. Transmit the modulated data through a Rayleigh fading channel.
5. Demodulate the noisy received data using BPSK or QPSK (coherent and noncoherent).
6. Feed the demodulated noisy signal into the AdaBoost algorithm.
7. Calculate the BER.
8. Plot the results of the BER vs. signal-to-noise ratio (SNR).

4. Simulation Results

After applying the AdaBoost algorithm, a performance comparison of different modulation techniques such as BPSK and QPSK is carried out. AdaBoost is used to improve the SNR of these techniques. The Algorithm is first applied to the coherent systems to improve the BER of the used modulation techniques for a Gaussian channel. It is next used to improve the BER of the noncoherent detection of the used modulation techniques in the AWGN and Rayleigh environments. A flowchart of the proposed system is shown in Figure 3.

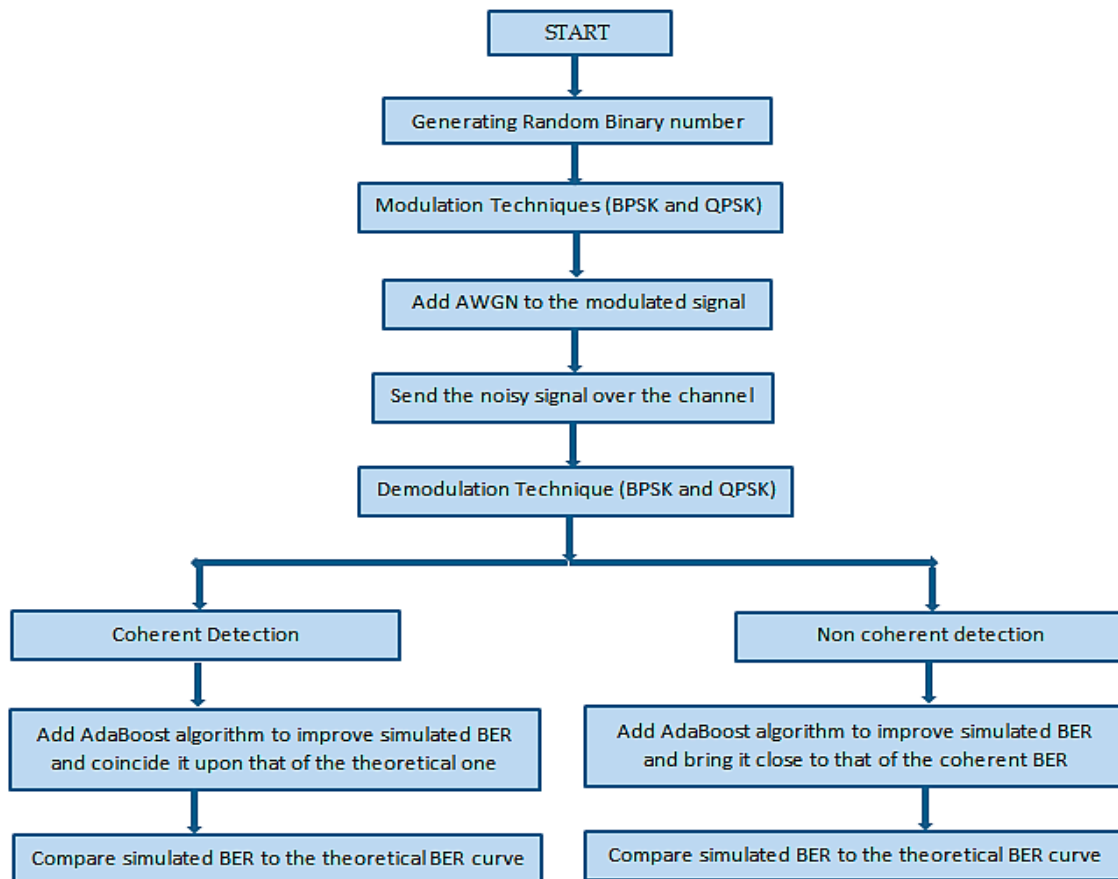


Figure 3. Flowchart of the proposed system. BPSK = binary phase shift keying; QPSK = quadrature phase shift keying.

In the following section, the MATLAB simulation of each technique is illustrated, which proves the improvement made by applying the AdaBoost to the recovered signal at the receiver. The recovered signal, still contaminated with some noise will pass through the AdaBoost algorithm for tuning of its parameters.

4.1. Binary Phase Shift Keying (BPSK)

4.1.1. Coherent BPSK

The model implements or simulates a Binary Phase Shift Keying (BPSK) system using MATLAB and obtains its BER vs. SNR. To evaluate the performance of the system, a simulation is carried out for 10^5 bits, and SNR equal to 0 to 9 dB as used in most of the simulations carried out for the BPSK system. An AWGN is added of variance equal to $1 / (10^{(SNR/10)})$, then, the noisy signal is sent over the Gaussian channel. After that, the AdaBoost algorithm is added at the receiver, where the observation weights $w_i = 1/n, i = 1, 2, \dots, n$, and n is the length of the input bits. The number of weak classifiers (T) used for BPSK is equal to 5 weak learners, where it is the number of trials after which the desired

result is achieved. The number of classes (K) is equal to 2 because in the case of BPSK, we are only interested in dividing the data into two classes. The AdaBoost technique separates the data into two classes achieving the BPSK concept as shown in Figure 4.

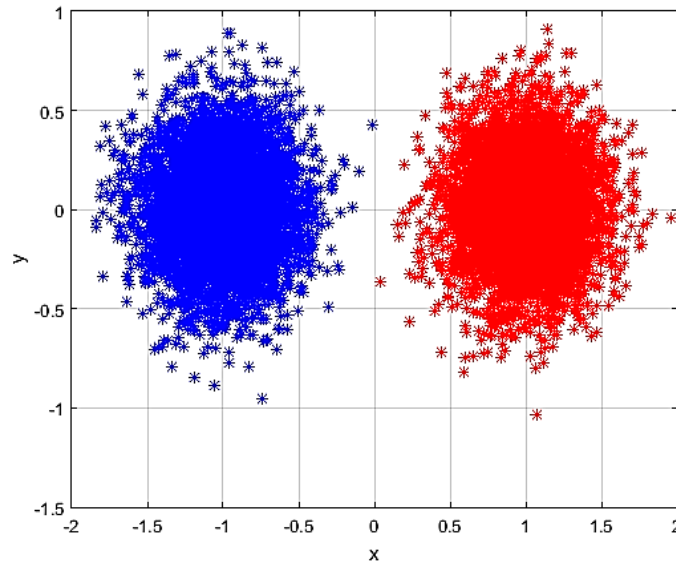


Figure 4. BPSK modulation technique.

Figure 5 shows the relation between both simulated and theoretical BER versus SNR of the coherent BPSK system. The results of simulated and analytical BER match very well. It is seen that as the value of the SNR increases, the corresponding value of the BER decreases. This makes sense since the SNR is equal to the power of the signal/power of the noise, so the value of the SNR increases. As a result, both curves coincide with each other. This is caused by adding the AdaBoost algorithm that helps in detecting the original data from the noisy received signal and reconstructing the constellation diagram and, as a result, the BER decreases. Hence, this verifies the improving effect of the AdaBoost algorithm on the used modulation technique.

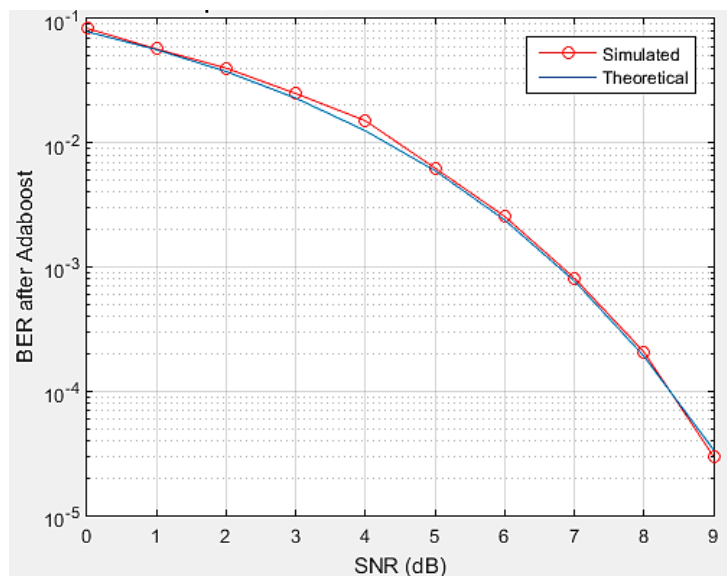


Figure 5. BER vs. SNR for coherent BPSK modulation over the AWGN channel.

4.1.2. Noncoherent BPSK

Moving on, the performance of the noncoherent BPSK system is evaluated in both the AWGN and Rayleigh environment along with the AdaBoost algorithm being added to improve the system BER.

- AWGN Channel

Figure 6 displays the BER vs. SNR in the case of the AWGN, where the simulated noncoherent system shows a better result than that of the theoretical. The AdaBoost algorithm improves the performance, specially at low SNR values. However, at high SNRs, the improvement decreases but still it is better than that of the theoretical as the overall curve is close in results to that of the coherent BPSK performance.

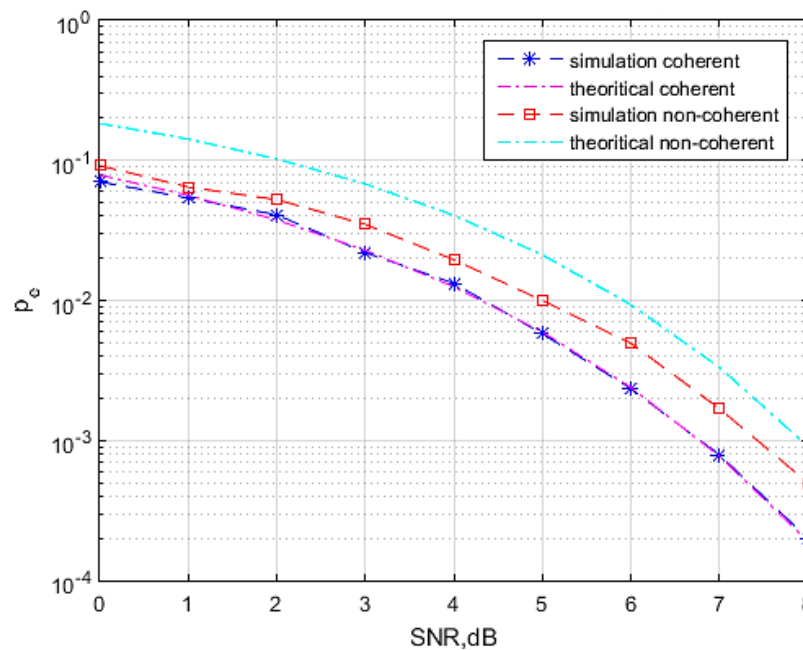


Figure 6. BER vs. SNR for noncoherent BPSK modulation over the AWGN channel.

As mentioned, the main effect of the AdaBoost algorithm is to recover the transmitted data from the noisy received signal, and hence, allow for reconstruction of the constellation diagram of the modulation technique. As a result, the simulated BER is improved. The improvement appears in the form of better results of the noncoherent simulated BER in comparison to that of the theoretical. Also, it can be seen that the noncoherent simulated curve approaches that of the coherent system. This is caused by the fact that the noncoherent system is already worse in performance than the coherent one, so when the AdaBoost algorithm is added, it improves its BER. Therefore, the AdaBoost algorithm is able to achieve a coherent performance for the noncoherent system.

- Rayleigh Fading Channel

Figure 7 displays the BER vs. SNR in the case of the Rayleigh fading environment. The related simulation parameters are $N = 10^5$ (no. of bits), $N_s = 5 \times 10^5$ (no. of symbols), and SNR = 0–20 dB. The Rayleigh fading channel parameters are given as follows: $f_c = 2 \times 10^9$ Hz, $c = 3 \times 10^8$ m/s, $v = 14$ m/s, $f_D = 93$ Hz, $D_s = 185$ Hz (Doppler spread), and $T_s = 1 \times 10^{-4}$ s (sampling time).

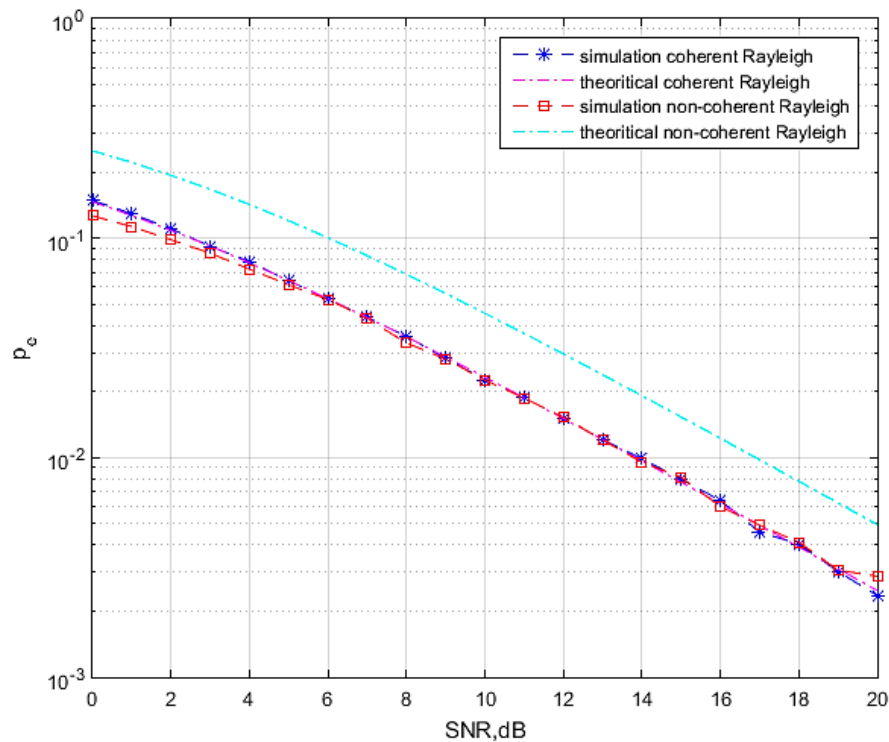


Figure 7. BER vs. SNR plot for noncoherent BPSK modulation over the Rayleigh fading channel.

The noncoherent system appears to be better than that of the theoretical noncoherent one since the performance of the system in the Rayleigh environment is already worse than that of the system with the AWGN only. It can be recognized that at a low SNR, the BER of the noncoherent system in the Rayleigh environment appears to be even lower than that of the coherent theoretical system of Rayleigh channel. As the SNR increases, the simulated noncoherent curve appears to almost follow the curve of both the simulated and theoretical coherent systems.

The same AdaBoost coherent performance achievement is also depicted in this figure. The AdaBoost algorithm manages to recover the transmitted data from the noisy signal received in the Rayleigh fading environment as well. Hence, it reconstructs the constellation diagram of the modulation technique. As a result, the simulated BER is improved. Again, when the AdaBoost algorithm is added, it improves its BER and achieves a coherent performance for the noncoherent system.

4.2. Quadrature Phase Shift Keying (QPSK)

4.2.1. Coherent QPSK

The MATLAB code simulates a QPSK system of 10^4 bits and SNR in the range of -3 to 7 in dB. An AWGN is generated with variance equal to $1/(10^{(SNR/10)})$ and is added to the transmitted signal. The same AdaBoost algorithm is used of $T = 15$, where it is the number of trials after which the desired result is achieved, and $K = 4$ because in the case of the QPSK, we are only interested in dividing the data into four classes. The algorithm separates the data into four classes achieving the QPSK concept as in Figure 8.

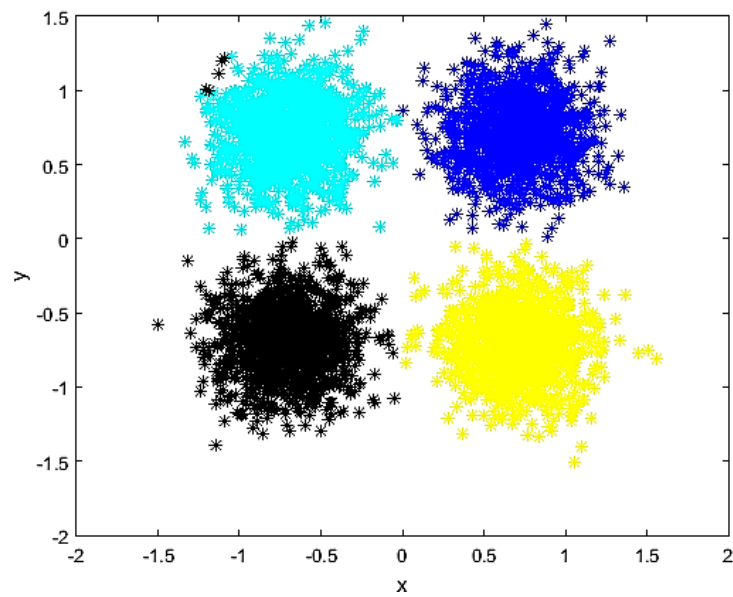


Figure 8. QPSK modulation technique.

Figure 9 illustrates the BER against the SNR of the coherent QPSK system. The figure depicts both simulated and theoretical results. At a low SNR, it is recognized that the BER increases, and at higher values of SNRs, the BER decreases. The simulated BER appears to almost coincide with the theoretical one except at the SNR value of 7 dB, where the simulated curve appears to dip slightly under the theoretical. The reason for this result is the same as that previously mentioned in discussion of Figure 6. Hence, this verifies the improving effect of the AdaBoost algorithm on the used modulation technique.

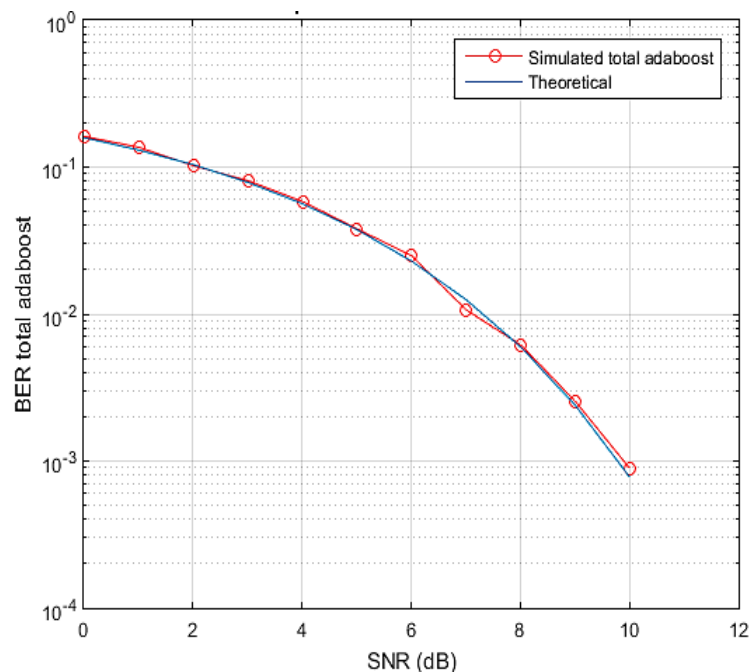


Figure 9. BER vs. SNR for coherent QPSK modulation over the AWGN channel.

4.2.2. Noncoherent QPSK

- AWGN Channel

Figure 10 shows the performance of the noncoherent QPSK in the case of the AWGN. The simulated system is of $N_s = 10^5$ (number of symbols) and SNR = 0–10 dB. An AWGN is added with variance = 1/(symbol power). A noncoherent detector is used to detect the received noisy signal. Then, the noisy signal goes through the AdaBoost algorithm of $T = 10$ (number of weak classifiers) and $K = 4$ (number of classes) to improve the BER curve.

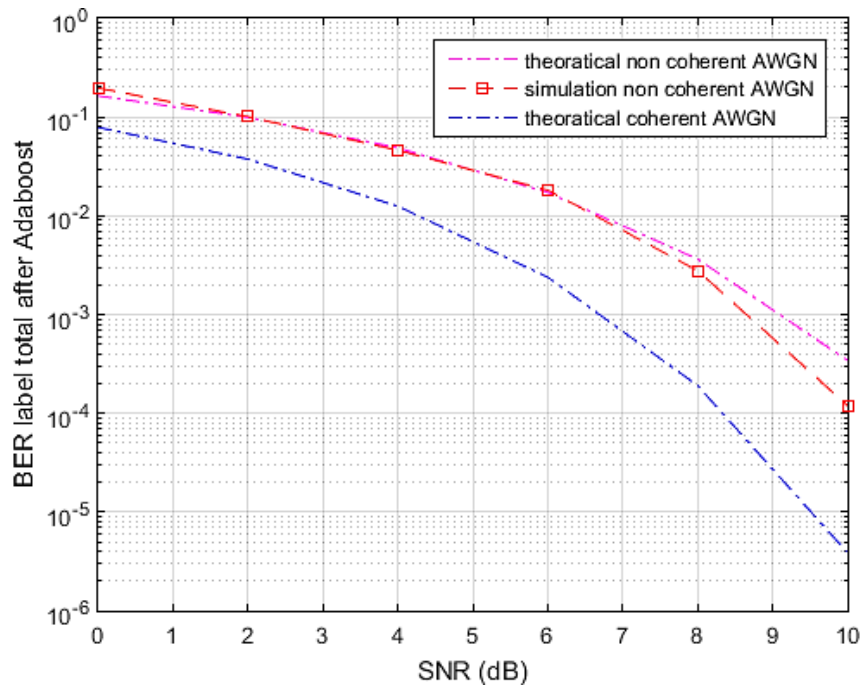


Figure 10. BER vs. SNR for noncoherent QPSK modulation over the AWGN channel.

The figure shows the theoretical and simulation curves of the noncoherent QPSK system. The simulated curve is improved over the theoretical one. At the beginning, the simulated curve appears to go slightly above the theoretical one. After that, it coincides with the theoretical curve up to a SNR of 7 dB. Then, starting from 7 dB and up to 10 dB, the simulated curve appears to improve as it goes under the theoretical one by about 0.00022. The improvement is caused by the added AdaBoost algorithm which helps in correcting the BER of the received noisy signal. This is done by the prediction capability of the algorithm that detects the original bit stream of the data sent.

- Rayleigh Fading Channel

In Figure 11, the performance of the noncoherent QPSK is evaluated by simulating a system of $N_s = 10^5$ (number of symbols) and SNR = 10–40 dB. An AWGN is added of variance = 1/(symbol power). The signal is sent through a Rayleigh channel of $f_D = 100$ Hz (Doppler frequency), $T_s = 1 \times 10^{-4}$ s (sampling time), and $c = 3 \times 10^8$ m/s. A noncoherent detector is used to detect the received noisy signal. Then, the noisy signal goes through the AdaBoost algorithm of $T = 5$ (number of weak classifiers) and $K = 4$ (number of classes) to improve the BER curve.

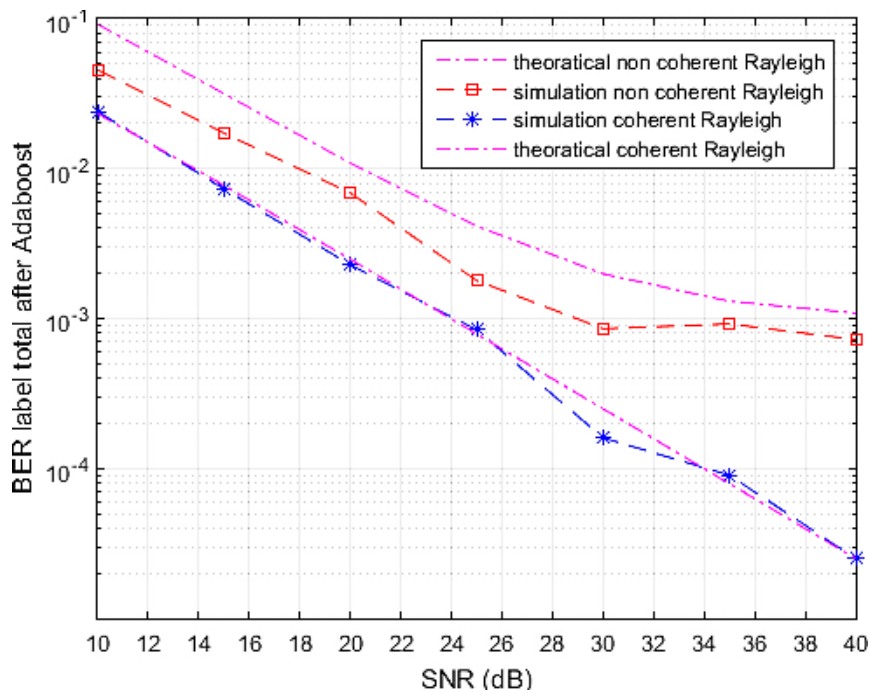


Figure 11. BER vs. SNR for noncoherent QPSK modulation over the Rayleigh fading channel.

Figure 11 shows the theoretical and simulation curves of the noncoherent QPSK system. The simulated curve is improved over the theoretical one. It appears to go under the theoretical curve by around 0.046 for SNR = 10–20 dB. Starting from SNR = 20–35 dB, the simulated curve goes under by 0.0040 to 0.0011 and then from SNR = 35–40 dB it goes under the theoretical curve by 0.0004. Again, and as stated before, the AdaBoost algorithm is able to achieve a coherent performance for the noncoherent system.

In comparison to the theoretical bit error rate (BER) vs. SNR curve, AdaBoost shows an improvement in the BER of coherent detection of BPSK and QPSK. The curve is coincided with that of each modulation technique, proving its improvement effect on the BER vs. SNR curve. As for the noncoherent system, the BER curve shows an improvement over that of the theoretical one. The AdaBoost algorithm improved BER curve and the BER curve became closer in result to that of the coherent detection. This result shows that by using the AdaBoost detection characteristics or capabilities, the value of the phase of the received signal could be calculated, which was unknown or highly noisy in the case of noncoherent detection.

4.3. Eight Quadrature Amplitude Modulation (8QAM)

The 8QAM coherent system is simulated for 30,000 bits and SNR values in the range of –6 to 10 dB. An AWGN of variance = $1/(10^{(SNR/10)})$ is added and then the bits are sent over a Gaussian link to the receiver. The received noisy signal then goes through the AdaBoost algorithm of $T = 25$ and $K = 8$, so as to separate the data into eight classes achieving the 8QAM concept as shown in Figure 12.

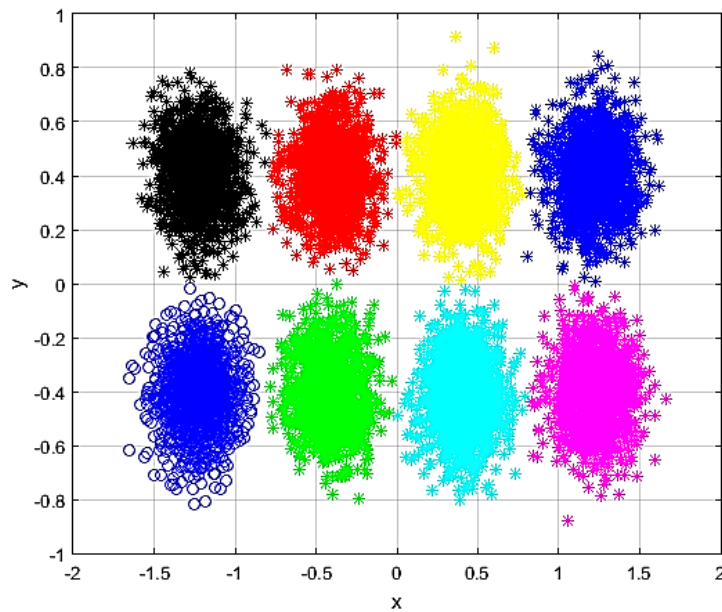


Figure 12. Eight Quadrature Amplitude Modulation (8QAM) modulation technique.

Figure 13 depicts both simulation and theoretical curves of BER vs. SNR of coherent 8QAM. It is noted that at low SNRs, the BER increases, while it decreases at higher SNRs. At the beginning of the curve, the simulated result is slightly higher than the theoretical one. At the end the simulated curve, it slightly dips under the theoretical curve; other than that, the two curves match very well.

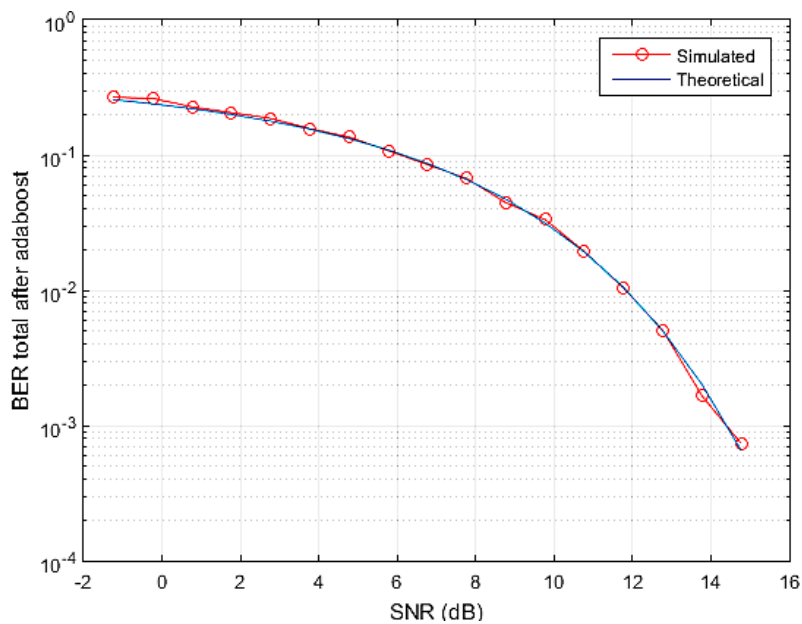


Figure 13. BER vs. SNR for coherent 8QAM modulation over the AWGN channel.

4.4. Sixteen Quadrature Amplitude Modulation (16QAM)

Similar to the last models, the coherent 16QAM system is simulated for 4×10^4 bits and SNR values from -6 to 10 dB. Figure 14 shows the 16QAM constellation after being separated into sixteen classes using the AdaBoost algorithm of $T = 35$ and $K = 16$.

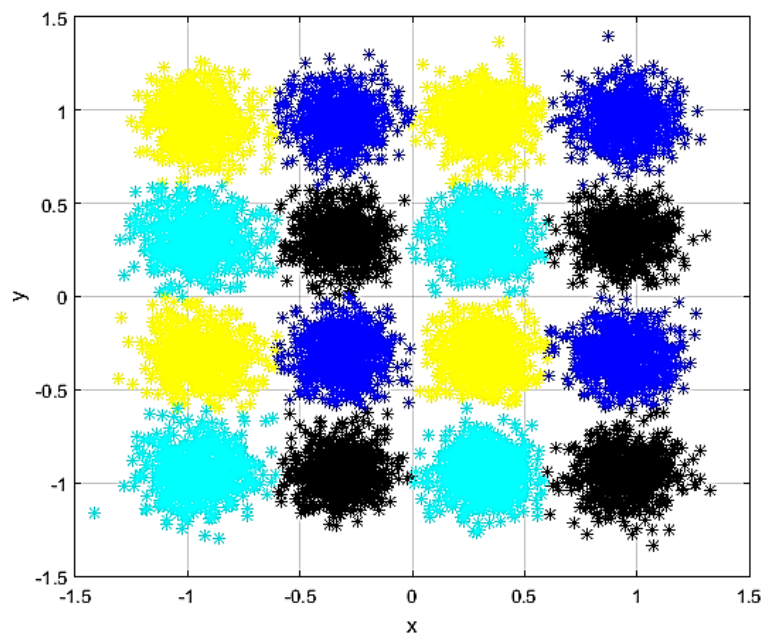


Figure 14. 16QAM modulation technique.

Figure 15 shows the BER vs. SNR curve for the 16QAM system, both simulated and theoretical. At low SNRs, the simulated curve results are slightly higher than the theoretical ones up to SNR values of 4 dB. After that, the simulated and analytical results seem to match very well, and the two curves seem to coincide with each other up to 16 dB.

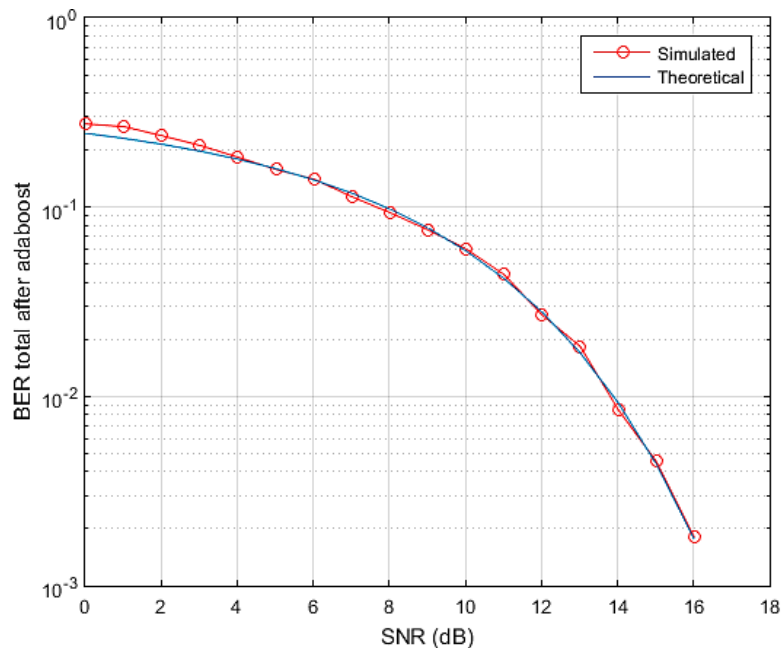


Figure 15. BER vs. SNR for coherent 16QAM modulation over the AWGN channel.

As a comparison with our proposed algorithm, BER curves are added for the performance of a noncoherent scheme using another machine learning (ML) algorithm. In [33], the research aims to handle the joint transmitter and noncoherent receiver optimization for multiuser single-input multiple-output (MU-SIMO) communications through unsupervised deep learning. It is revealed that MU-SIMO can be modeled as a deep neural network with three vital layers, which includes a partially-connected linear layer for joint multiuser waveform design at the transmitter side, and two

nonlinear layers for the noncoherent signal recognition. The suggested method shows a remarkable MU-SIMO noncoherent communication performance in Rayleigh fading channels. The results are shown in the following figures. Figure 16 shows the block error rate (BLER) as a function of E_b/N_0 for the single-input single-output (SISO) case. As for Figure 17, it displays the BLER as a function of E_b/N_0 for the MU-SIMO case.

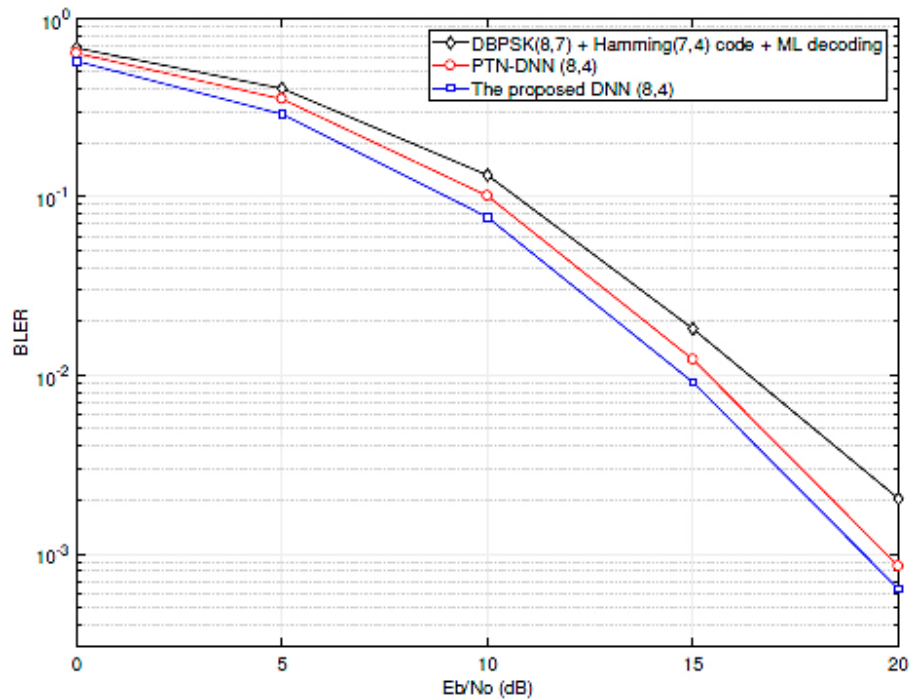


Figure 16. Block error rate (BLER) as a function of E_b/N_0 for the single-input single-output (SISO) case.

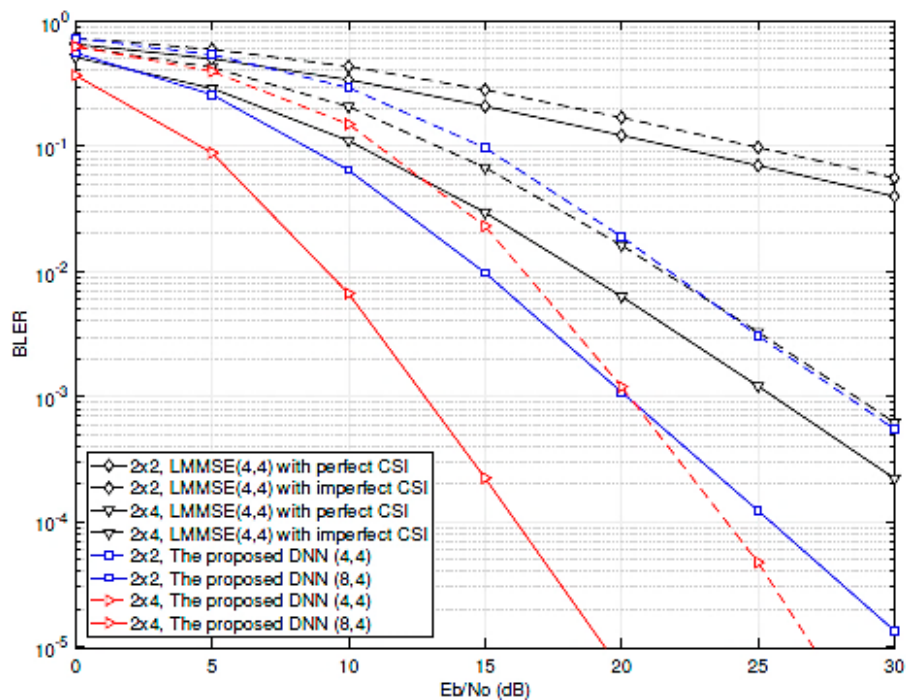


Figure 17. BLER as a function of E_b/N_0 for the multiuser single-input multiple-output (MU-SIMO) case.

5. Conclusions

The AdaBoost algorithm is used to improve the SNR of different modulation techniques. It shows an improvement in the SNR of BPSK and QPSK. In comparison to the theoretical BER vs. SNR curve, the AdaBoost curve coincides with that of each modulation technique. A BER similar to that of the theoretical coherent one is achieved for each modulation technique without adding further complexity to the system. The AdaBoost algorithm is next used to improve the BER of the noncoherent detection of the used modulation techniques. AdaBoost helps in improving the noncoherent BER by detecting and calculating the value of the phase of the received bits. The achieved BER shows improvement over that of the theoretical noncoherent one. It yields a BER closer to that of the coherent detection. This result shows that by using the AdaBoost detection characteristics or capabilities, the value of the phase of the received signal could be calculated, which was unknown or highly noisy in the case of noncoherent detection. This helps to improve the noncoherent detection of the receiver without adding further complexity, which was originally the advantage (i.e. lower complexity) of the noncoherent over the coherent detection. The BER of the noncoherent detection is improved, which was the tradeoff versus that of the coherent detection. These improvements are achieved for the BPSK and QPSK noncoherent systems in both AWGN and Rayleigh fading channels.

Author Contributions: Conceptualization, M.E.K. and M.R.M.R.; Methodology, H.G.; Software, H.G.; Validation, H.G., M.E.K. and M.R.M.R.; Formal Analysis, H.G.; Investigation, H.G.; Resources, M.E.K.; Data Curation, H.G.; Writing-Original Draft Preparation, H.G.; Writing-Review & Editing, M.E.K., M.R.M.R., M.H.A. and N.E.I.; Visualization, H.G. and M.E.K.; Supervision, M.E.K., M.H.A., N.E.I. and M.R.M.R.; Project Administration, M.H.A., N.E.I. and M.R.M.R.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Zhu, J.; Zou, H.; Rosset, S.; Hastie, T. Multiclass AdaBoost. *Stat. Its Interface* **2009**, *2*, 349–360. [[CrossRef](#)]
- Freund, Y.; Scapire, R.E. A short introduction to boosting. *JSAI* **1999**, *14*, 771–780.
- Freund, Y.; Scapire, R.E. A decision-theoretic generalization of online learning and an application to boosting. *J. Comput. Syst. Sci.* **1997**, *55*, 119–139. [[CrossRef](#)]
- Freund, Y.; Scapire, R.E. Experiments with a new boosting algorithm. In Proceedings of the 13th International Conference on Machine Learning (ICML'96), Bari, Italy, 3–6 July 1996; pp. 148–156.
- Binder, H.; Gefeller, O.; Schmid, M.; Mayr, A. The evolution of boosting algorithms. *Methods Inf. Med.* **2014**, *53*, 419–427. [[CrossRef](#)] [[PubMed](#)]
- Breiman, L. Arcing classifiers (with discussion). *Ann. Stat.* **1998**, *26*, 801–849. [[CrossRef](#)]
- Breiman, L. Prediction games & arcing algorithms. *Neural Comput.* **1999**, *11*, 1493–1517. [[CrossRef](#)]
- Friedman, J.; Hastie, T.; Tibshirani, R. Additive logistic regression: A statistical view of boosting (with discussion). *Ann. Stat.* **2000**, *28*, 337–407. [[CrossRef](#)]
- Ratliff, N.D.; Silver, D.; Bagnell, J.A. Learning to search: Functional gradient techniques for imitation learning. *Auton. Robots* **2009**, *27*, 25–53. [[CrossRef](#)]
- Wu, S.; Nagahashi, H. Penalized AdaBoost: Improving the generalization error of gentle AdaBoost through a margin distribution. *IEICE Trans. Inf. Syst.* **2015**, *E98-D*, 1906–1915. [[CrossRef](#)]
- Jordan, M.I.; Mitchell, T.M. Machine learning: Trends, perspectives, and prospects. *Science* **2015**, *349*, 255–260. [[CrossRef](#)]
- Zhang, Y.; Sow, D.; Turaga, D.; van der Schaar, M. A fast online learning algorithm for distributed mining of bigdata. *ACM Sigmetrics* **2014**, *41*, 90–93. [[CrossRef](#)]
- Linero, A.R. Bayesian regression trees for high-dimensional prediction and variable selection. *J. Am. Stat. Assoc.* **2018**, *113*, 626–636. [[CrossRef](#)]
- Ridgeway, G. Looking for lumps: Boosting and bagging for density estimation. *Comput. Stat. Data Anal.* **2002**, *38*, 379–392. [[CrossRef](#)]
- Wu, K.; Hou, W.; Yang, H. Density estimation via the random forest method. *Commun. Stat. Theory Methods* **2018**, *47*, 877–889. [[CrossRef](#)]

16. Wang, H.; Wang, J.; Zhou, L. A survival ensemble of extreme learning machine. *Appl. Intell.* **2018**, *48*, 1846–1858. [[CrossRef](#)]
17. Miller, P.J.; Lubke, G.H.; McArtor, D.B.; Bergeman, C.S. Finding structure in data using multivariate tree boosting. *Psychol. Methods* **2016**, *21*, 583–602. [[CrossRef](#)] [[PubMed](#)]
18. Lu, K.; Zhang, W.; Sun, B. Multidimensional Data-Driven Life Prediction Method for White LEDs Based on BP-NN and Improved-Adaboost Algorithm. *IEEE Access* **2017**, *5*, 21660–21668. [[CrossRef](#)]
19. Seibold, H.; Bernau, C.; Boulesteix, A.-L.; De Bin, R. On the choice and influence of the number of boosting steps for high-dimensional linear cox-models. *Comput. Stat.* **2017**, *33*, 1195–1215. [[CrossRef](#)]
20. Liu, T.; Guan, Y.; Lin, Y. Research on modulation recognition with ensemble learning. *EURASIP J. Wirel. Commun. Netw.* **2017**, *179*, 1–10. [[CrossRef](#)]
21. Ben Chikha, W.; Chaoui, S.; Attia, R. Performance of AdaBoost classifier in recognition of superposed modulations for MIMO TWRC with physical-layer network coding. In Proceedings of the 25th International Conference on Software, Telecommunications and Computer Networks (SoftCOM2017), Split, Croatia, 21–23 September 2017; pp. 1–5.
22. Khieu, H.-T.; Le, D.-K.; Nguyen, B.V. On the performance analysis of a DCSK system under the pulse jamming environment. *PLoS ONE* **2018**, *13*, 1–11. [[CrossRef](#)]
23. Herceg, M.; Kaddoum, G.; Vranjes, D.; Soujeri, E. Permutation Index DCSK Modulation Technique for Secure Multiuser High-Data-Rate Communication Systems. *IEEE Trans. Veh. Technol.* **2018**, *67*, 2997–3011. [[CrossRef](#)]
24. Nguyen, B.V.; Jung, H.; Kim, K. On the Antijamming Performance of the NR-DCSK System. *Secur. Commun. Netw.* **2018**, 1–9. [[CrossRef](#)]
25. Kaddoum, G.; Tran, H.-V.; Kong, L.; Atallah, M. Design of Simultaneous Wireless Information and Power Transfer Scheme for Short Reference DCSK Communication Systems. *IEEE Trans. Commun.* **2016**, *65*, 431–443. [[CrossRef](#)]
26. Kumar, M.A.; Babu, R.S. A Novel SUI Modelling for Multiuser OFDM-DCSK Modulation based Communication System. *Int. J. Adv. Eng. Res. Dev.* **2017**, *4*, 345–356. [[CrossRef](#)]
27. Schapire, R.E. A brief introduction to boosting. In Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI'99), Stockholm, Sweden, 31 July–6 August 1999; pp. 1401–1406.
28. Tang, J.; Deng, C.; Huang, G.-B. Extreme learning machine for multilayer perceptron. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *27*, 809–821. [[CrossRef](#)] [[PubMed](#)]
29. Schapire, R.E.; Freund, Y.; Bartlett, P.; Lee, W.S. Boosting the margin: A new explanation for the effectiveness of voting methods. *Ann. Stat.* **1998**, *26*, 1651–1686. [[CrossRef](#)]
30. Ahmadi, N.; Berangi, R. A template matching approach to classification of QAM modulation using genetic algorithm. *Int. J. Signal Process.* **2009**, *3*, 95–109.
31. Li, P.; Zhang, Z.; Wang, X.; Xu, N.; Xu, Y. Modulation recognition of communication signals based on high order cumulants and support vector machine. *J. China Univ. Posts Telecommun.* **2012**, *19*, 61–65. [[CrossRef](#)]
32. Jain, A.K.; Duin, P.W.; Mao, J. Statistical pattern recognition: A review. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 4–37. [[CrossRef](#)]
33. Xue, S.; Ma, Y.; Yi, N.; Tafazolli, R. Unsupervised Deep Learning for MU-SIMO Joint Transmitter and Noncoherent Receiver Design. *IEEE Wirel. Commun. Lett.* **2018**, 1–4. [[CrossRef](#)]

