5-12-2017

Symbol Synchronization Techniques in Digital Communications

Mohammed Al-Hamiri
mga5528@rit.edu

R·I·T

Symbol Synchronization Techniques in Digital Communications

by

Mohammed Al-Hamiri

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Telecommunications Engineering Technology

Electrical, Computer and Telecommunications Engineering Technology

College of Applied Science and Technology

Rochester Institute of Technology

Rochester, NY

(May 12, 2017)

Committee Approval

Miguel Bazdresch Date

Thesis Advisor / Professor

William P. Johnson Date

Committee Member / Professor

Steven Ciccarelli Date

Committee Member / Associate Professor

Abstract

Timing synchronization plays an important role in recovering the original transmitted signal in telecommunication systems. In order to have a communication system that operates at the correct time and in the correct order, it is necessary to synchronize to the transmitter's symbol timing. Synchronization can be accomplished when the receiver clock tracks the periodic timing information in a transmitted signal to reproduce the original signal.

In this thesis work, we report the design, implementation and evaluation of a timing synchronization algorithm based on the technique first proposed by Gardner [1], applied to wireless communication using the Alamouti space-time code [2] under QPSK modulation with half-sine pulses. To achieve this, a mathematical model is introduced which includes software design of communication algorithms. In this modeling, we simulate the Gardner algorithm in MATLAB. Then, five techniques are introduced to improve the performance of the loop filter in the digital receiver, and they are successfully implemented and evaluated in Matlab. These five techniques prove that there is an improvement in digital receiver performance in terms of the convergence speed and the communication system complexity.

On the other hand, the optimum decoding of the Alamouti space-time code, as initially proposed, makes the non-trivial assumption that the communication system is perfectly synchronized. Realistic wireless environments contain additive white Gaussian noise (AWGN), multipath fading, and it is not perfectly synchronized. In this thesis, the Alamouti space-time code technique is written for QPSK modulation scheme to work in realistic environment that involves a timing synchronization technique. We compare the bit error rate (BER) of the Alamouti decoder when synchronized using the proposed algorithms with the ideal results found in the literature, and

we find them to be similar, proving that the synchronization algorithm is in fact achieving optimum synchronization.

This thesis presents synchronization algorithms that are necessary for a complete working wireless-Alamouti technique. Also, this thesis improves the communication system performance in terms of the convergence speed with reducing the computational complexity of the communication system design.

Table of Contents

Chapter 1

Introduction

1.1 Background

Synchronization is the process of technical coordination between transmitters and receivers in digital communication systems. Kihara, Ono, and Eskelinen [3] show that synchronization is required for fast and reliable data transfer from transmitters to receivers, and to enable every individual component to be synchronized wherever the component is placed in digital communication system. Transmitters and receivers must be mutually coordinated to transfer data successfully. The receiver accepts data as true information only when the receiver knows the digital clock of the transmitter, or when the receiver has the ability to regenerate the digital clock. So, it is impossible to have a communication system that properly works without using synchronization in communication devices [3].

According to the synchronization level, there are two types of digital communication systems: asynchronous and synchronous systems. In asynchronous systems, local clock synchronization is established; whereas, all clocks are completely bound together in synchronous systems. Kihara, Ono, and Eskelinen [3] state that, "Asynchronous clocks are assumed to be independent and no effort is made to force them to synchronism. Of course, here the clocks are synchronized in practice to some extent."

In general, both the synchronous and asynchronous systems are used in the current digital communication systems. Figure (1.1) shows the block diagram of the digital receiver that contains two low pass filters, and a one matched filter. In addition, it involves a carrier synchronization, symbol synchronization, and frame synchronization. The focus of this thesis is on the symbol synchronization. The term of symbol synchronization or the timing recovery has the same meaning in digital communication systems.

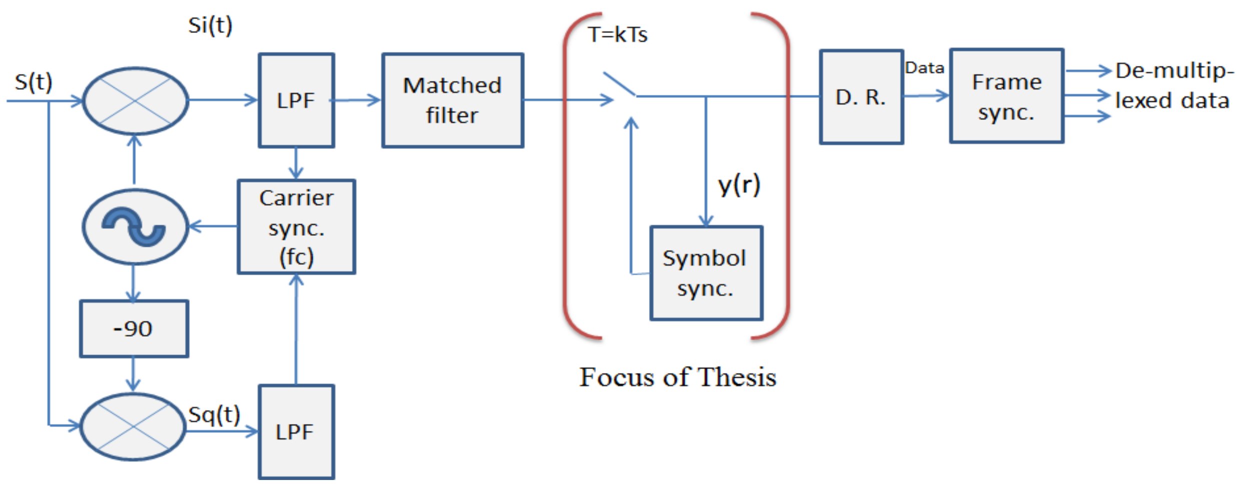

Figure 1.1: Receiver block diagram

1.2 Review of past studies and their limitations

In digital receivers, the timing recovery leads to obtain symbol synchronization. Floyd Gardner [1] states that timing adjustment can be achieved by interpolation if the sampling is not synchronized to the data symbols. Some of proposed solutions are to use synchronous Double Side Band systems. Costas [4] shows that Double Side Band has power advantage over Single Side Band when all factors, such as system complexity and susceptibility to jamming, are taken into account. Franks [5] illustrates that Maximum-likelihood estimation theory is another solution for the timing recovery which depends on root mean square jitter of the timing parameters as an approach to the evaluation of timing recovery circuit performance.

C. R. Johnson, Jr. and W. A. Sethares [6] demonstrate that the problem of clock recovery can be described by finding a timing offset. When a timing offset is accurately detected, the energy of the received signal can be maximized. This problem can be solved by a linear optimization technique such as gradient descent, which leads to a standard algorithm for timing recovery.

The previous studies focus on the synchronization techniques, but these studies ignored other important factors in designs of digital communication systems. In other words, these studies depend in their working on adding new components to the basic communication system. These components are added to detect the error in the synchronization process, such as error detection components, to manipulate the multipath fading, and to remove the noise that comes from external environment, such as filters. As a result, adding these components to the communication system leads to increase its complexities. These complexities can be described by the delay and the computational complexity that are produced by each additional component in wireless communication systems.

In terms of the delay, each component needs a time to conduct its required functions which is called the processing time. As a result, these additional components cause a delay for the wireless communication system. This delay has negative effects on the spectrum efficiency in term of the Bandwidth (BW). Also, this delay leads to decrease the speed of the convergence for the wireless communication system.

1.3 Purpose statement

The purpose of this thesis is to generate new algorithms that simplify the communication system complexities and improve the performance of digital communication systems. This can be accomplished by using a modulation scheme which is called Quadrature Pulse Shift Keying (QPSK) with two transmitters and a one receiver (the Alamouti technique). Goldsmith [7] states that the QPSK modulation scheme has the ability to encode two bits per symbol which increases data rates and improves the spectrum efficiency.

In addition, the Alamouti technique [2] can be used to reduce the bit error rate by depending on the diversity improvements. The diversity technique has positive effects on the reception quality by reducing the fading effects. As a result, the Alamouti technique is used to reduce the required components to recover the original signal. Consequently, using QPSK modulation scheme with the Alamouti technique contribute in improving the performance of the digital communication system.

1.4 Hypotheses

This study aims to generate new algorithms and investigate if the new algorithms improve the digital communication system performance in term of the convergence speed with reducing the complexities of the communication system design.

H1: The new algorithms improve the digital communication system performance in term of the convergence speed with reducing the complexities of the communication system design.

H2: The new algorithms improve the digital communication system performance in term of the convergence speed without reducing the complexities of the communication system design

H3: The new algorithms improve the digital communication system performance by reducing the complexities of the communication system design without any improvement in the convergence speed.

H4: The new algorithms do not improve the digital communication system performance in term of the convergence speed and the complexities of the communication system design.

Chapter 2

Literature Review

This literature review introduces information about the QPSK modulation scheme, the symbol synchronization, timing recovery techniques, the Gardner technique, and the Alamouti space-time code technique.

2.1 Quadrature Phase Shift Keying (QPSK)

QPSK is a digital modulation technique that sends two bits per symbol, and each symbol carries one of the four possible bit combinations (00, 01, 10, or 11). The phase of the carrier varies according to the symbol, and there are four phase shifts. The receiver needs to recover the original information from the modulated signal. QPSK is a bandwidth efficient because it sends two bits per symbol. That can be shown by comparison between QPSK and the Binary Phase Shift Keying (BPSK).

BPSK sends one bit per symbol because it uses two possible phase shifts. So, the baseband signal has a certain frequency that can be used to send one bit per each symbol period. In the QPSK case, the baseband signal, that has the same frequency above, can be used to send two bits per each symbol period. As a result the bandwidth efficiency of QPSK is higher by a factor of two [8]. In other words, the frequency spectrum that is needed to transmit data by using QPSK modulation scheme is the half of that required to transmit the same amount of data by using BPSK modulation scheme. The transmitted symbols are represented by complex numbers, so the first bit is the real part and the second bit is the imaginary part. The following table demonstrates the encoding rules that are used to represent the four possible phase shifts for QPSK modulation scheme.

The phase options	The complex numbers
00	-1-1j
01	-1+1j
10	1-1j
11	1+1j

Table 2.1: This table shows the encoding rules to represent the four phase options as complex numbers.

2.2 Symbol synchronization

The digital received signal passes through several components in the digital receiver which contribute to the recovery of the original transmitted signals [7]. One of these components is the digital demodulator, which is responsible for the acquisition of accurate symbol timing. Timing information that is obtained from synchronization is useful to delineate the digital received signal that is associated with a given symbol. Sampling techniques depend upon the timing information for amplitude, phase, and frequency demodulation [7].

The principle of timing synchronization depends on estimation of the timing offset τ of the signal in AWGN channels. Digital wireless receivers face difficulties in estimating the value of τ due to the time-varying multipath in addition to the noise. Strode and Groves [9] state that τ can be distorted by the time-varying multipath and the noise. Goldsmith [7, p. 161] mentions that, "In most performance analysis of wireless communication systems it is assumed that the receiver synchronizes to the multipath component with delay equal to the average delay spread". After this, the channel will be considered as AWGN to estimate the symbol timing.

In the symbol synchronization, the synchronizer takes samples for the received signal. As a result, these samples will be used to acquire symbols. Mueller and Muller [10] suggest timing algorithms that depend on just one sample per symbol and require directed-decision operations. In carrier system, correct decisions depend on carrier phase that should be known previously. As a result, this carrier phase will indicate the value of symbol timing which will result in increasing the system complexities.

Suzuki et al. [11] propose a different scheme which is called Wave Difference Method (WDM). This method "finds the average location of zero-slope of the received signal filtered signal pulses" [11]. Actually, this method requires numerous samples per symbol which may result in increasing the processing time. Therefore, this method increases the demand on the bandwidth (BW) which means decreasing the spectrum efficiency. Agazzi et al. [12] suggest using the previous method which is the Wave Difference Method with only two samples per symbol. This method works only with baseband signals which represent the demodulator output.

2.2.1 The problem of Timing Recovery

The problem of timing recovery is the difficulty finding the optimal time for sampling [6]. This problem can be mathematically explained by finding the timing offset τ which the parameter that maximizes or minimizes some function of τ such as the output power or the cluster variance. When samples are taken, the τ parameter will be detected. Consequently, the output of the sampler will be a function of the τ parameter [6]. This can be showed by the following formula which states the baseband waveform at the input of the sampler:

$$x(t) = \sum_{i=-\infty}^{\infty} s[i]\,\delta(t - iT) * g_T(t) * c(t) * g_R(t) + w(t) * g_R(t).$$

where $s[i]$ represents the transmitted data, $g_T(t)$ is the pulse shaping filter, $c(t)$ is the impulse response of the channel, $g_R(t)$ is the receiver filter, and $w(t)$ is the noise. The three linear filters can be combined:

$$h(t) = g_T(t) * c(t) * g_R(t).$$

At ($kT/M + tau$) where M is the oversampling factor and T represents the interval between symbols, the sampled output can be written as:

$$x(kT/M + tau) = \sum_{i=-\infty}^{\infty} s[i]\,h(t - iT) + w(t) * g_R(t) \prod_{t=kT/M+tau}.$$

When the noise is supposed to have the same distribution whenever the samples are taken, the variance of the noise at the sampling time can be found without depending on the value of tau:

$$v(k) = w(t) * g_R(t) \prod_{t=kT/M+tau}.$$

By maximizing and minimizing some of the function of the samples, the value of tau can be found:

$$x(k) = x(kT/M + tau) = \sum_{i=-\infty}^{\infty} s[i]\,h(KT/M + tau - iT) + v(k).$$

Figure 2.1 shows algorithms of the timing recovery that can be implemented by three ways. C. R. Johnson, Jr. and W. A. Sethares [6] state that the first way can be done by using a digital post processor that indicates where the samples should be taken. The second way can be achieved by using an analog processor which determines when to sample. The third way can be accomplished by using a free running clock that chooses the sampling instants.

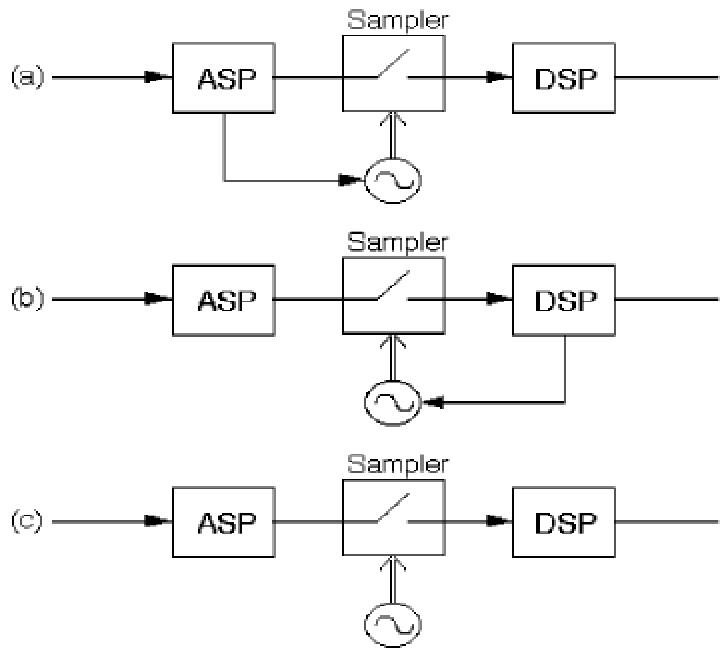

Figure 2.1: Three common structures for timing recovery. (a) Shows an analog processor. (b) States a digital post processor. (c) Use a free running clock and a digital post processor [6].

2.3 Timing Recovery Techniques

The signal is sent from the transmitter through the channel to the receiver, and the signal suffers due to the channel conditions. These conditions include many things such as noise, fading, and attenuation. The received signal is a complicated continuous waveform that needs to be treated before the sampling process. The sampling process is important to recover the transmitted signal which requires that samples should be taken at the optimal time [6, Ch. 12, p. 226]. The best time to take the samples is at the peak of the signal where the eye is opened widest.

The problem of timing recovery is to find the features of performance functions at the optimal time. These performance functions are employed to indicate the adaptive elements which are responsible about estimation the sampling times [6, Ch. 12, pp. 226]. If the sampling times are incorrectly taken away from the optimal time, this will result in an error which is called the source recovery error. This source recovery error represents the error between the transmitted data and the received data. C. R. Johnson, Jr. and W. A. Sethares [6] state that the source recovery error can be calculated only when there is a training sequence or when the transmitted data are known.

Another way to estimate the error between the transmitted data and the received data is by using the cluster variance. The cluster variance suggests taking the square between the nearest element of the source alphabet and values of the received data [6]. The measurement of the power of the *T*-spaced output of the matched filter is another approach to estimate the error. The estimation can be done by maximizing this output power which results in defining the adaptive element that is necessary to find the optimal time [6].

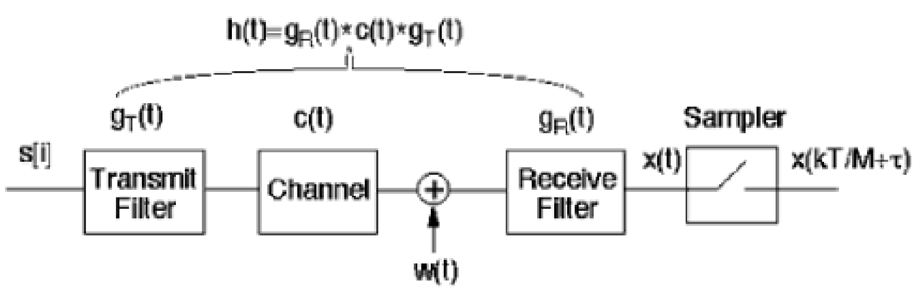

Figure 2.2: The effects of the transmitter pulse shaping g_T, the channel *c,* and the receive filter g_R can be represented by the transfer function *h* [6].

The various performance functions can be understood by drawing the error surfaces. In many situations, when the error surface for the output power is maximum, the error surface for the cluster variance is minimum. In these situations, methods of timing recovery can use either output power method or the cluster variance method as a basis to find the optimal time [6]. The quality of the timing offset τ can be measured by using the cluster variance which is explained in the following section.

2.3.1 Cluster Variance

The decision device $Q(x[k])$ quantizes the binary data to the number that is closed to it [6]. In other words, the decision device converts any negative value to -1, and any positive value to +1. In the case of the timing offset is larger than -T/2 and smaller than T/2, the eye is open, and the $Q(x[k]) = S[k-1]$ for all k, and the source recovery error can be represented by:

$$e[k]=s[k-1] - x[k]= Q(x[k])- x[k].$$

If the timing offset is smaller than -T/2 or larger than T/2, the $Q(x[k])$ will not be equal to $s[k-1]$. The Cluster Variance can be written as:

$$CV= avg\{e^2[k]\}= avg\{(Q(x[k]) - x[k])^2\}.$$

The Cluster Variance is a function of τ and the following figure shows the periodic nature of the function. C. R. Johnson, Jr. and W. A. Sethares [6] state that the problem of the timing recovery can be represented by a one dimensional search for the timing offset that minimizes the cluster variance.

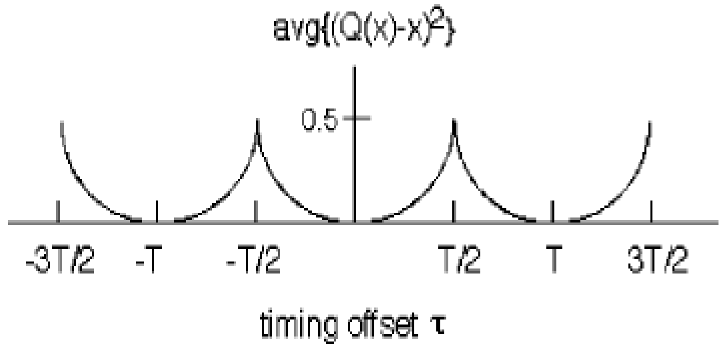

Figure 2.3: Cluster variance as a function of timing offset τ [6].

C. R. Johnson, Jr. and W. A. Sethares [6] state that the previous measure is applied in the simple case. The simple case includes a channel without noise, the pulse shape *h(t)* is a triangle pulse, the transmission is binary data, and there is no inter symbol interference ISI. In this ideal situation, the timing offset τ can be indicated either by maximizing the output power or by minimizing the cluster variance [6]. There are two methods that can be used to design adaptive elements that are responsible about the maximization or the minimization.

2.3.2 Timing Recovery via Decision-Directed Methods

The value of the waveform can be the same as the value of the transmitted data when the samples are taken at the optimal times correctly, and the pulse shape, the channel, and the matched filter are working properly. The best performance can be obtained by finding the sampling times that make the source recovery error as small as possible. The source recovery error is measured by finding the difference between the received data and the transmitted data when there is a training sequence.

It is not possible to calculate the source recovery error in the normal situation when there is no a training sequence. So, the timing recovery algorithm for the simple case cannot be applied in the normal situation. To solve this problem, C. R. Johnson, Jr. and W. A. Sethares [6] derived an algorithm to find the optimal value:

$$\tau[k+1] = \tau[k] + \mu(Q(x[k]) - x[k])\,[x(kT/M + \tau[k] + \delta) - x(kT/M + \tau[k] - \delta)].$$

where the μ is the step size. The step size can be reduced, or the numbers of the average values can be increased to eliminate the effect of the noise on the value of $[k]$. It is true that these two ways can reduce the influence of the noise, but they will slow the convergence of the algorithm. The algorithm above requires three samples for each symbol from the waveform, but it can be implemented easily. This can be done by taking samples three times straightforwardly [6]. The sampling is a hardware intensive solution because it involves hardware to achieve the sampling process.

The sampling theory states that any signal can be correctly reconstructed if it is sampled faster than twice the frequency. As a result, the value of the signal at $x(kT/M + \tau)$ can be used to interpolate and find the value of the signal at $x(kT/M + \tau[k] + \delta)$ and at $x(kT/M + \tau[k] - \delta)$ [6]. As a result, this section introduces a mathematical model to find the timing offset τ that minimize the cluster variance. The following figure states how the samples are taken which can be achieved by hardware.

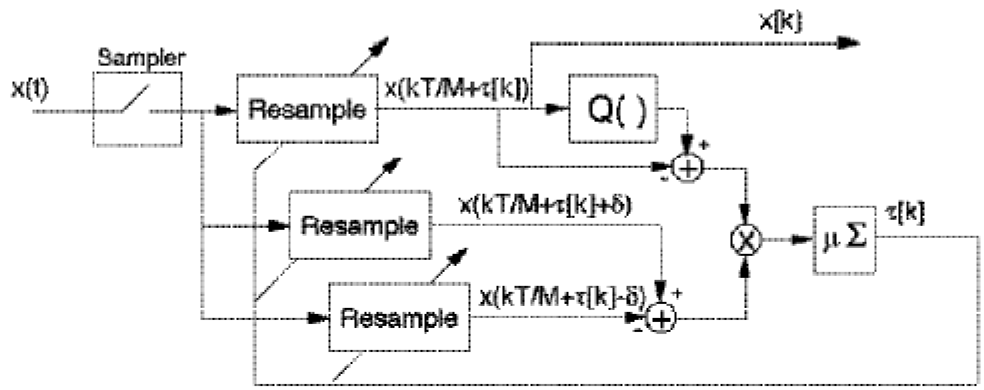

Figure 2.4: The adaptive element involves three interpolators. After the convergence of *τ[k],* the *x[k]* has the samples taken at times that minimize the cluster variance [6].

2.3.3 Timing Recovery via Output Power Maximization

The goal of timing recovery algorithm is to find the optimal time for sampling. This sampling can be done by maximizing the average of the received power (avg{x^2[k]}). This approach gives the same result that can be achieved by minimizing the cluster variance. This approach introduces an element that adapts τ to find the optimal time that maximizes the output power. C. R. Johnson, Jr. and W. A. Sethares [6] derived an algorithm to find the optimal value:

$$\tau[k+1]= \tau[k] + \mu\, x[k])\ [x(kT/M + \tau[k] + \delta) - x(kT/M + \tau[k] - \delta)].$$

The μ is the step size. If the μ is decreased, this can reduce the effect of the noise. Also, the effect of the noise can be eliminated by reducing the number of element that is used to find the average. However, this leads to reduce the speed of the convergent plot. This algorithm is a software intensive solution, and it can be showed by the next figure.

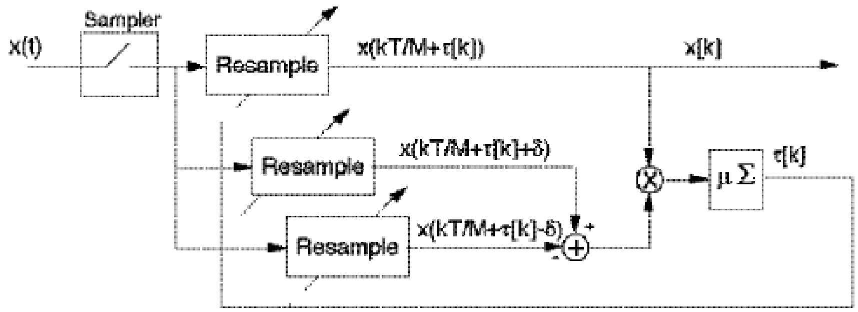

Figure 2.5: The adaptive element involves three interpolators. After the convergence of *τ[k],* the *x[k]* has the samples taken at times that maximize the output power [6].

The value of *x(t)* can be reconstructed at (*x(kT/M + τ[k] + δ)])* and at *x(kT/M + τ[k] – δ)]* from x[k]. This timing recovery algorithm can be implemented in digital, hybrid, and analog form [6].

As a result, these algorithms present solutions for the timing recovery which result from the mismatching between the receiver and transmitter clock. In addition to the timing recovery algorithms explained above that are suggested by C. R. Johnson, Jr. and W. A. Sethares [6], there are other techniques that are usually used for timing recovery. These techniques are the Mueller and Muller technique, the early-late technique, band-edge timing synchronization technique, and the Gardner technique.

2.3.4 Gardner technique

Gardner [1] proposes algorithms for timing error detector for limited band Binary Phase Shift Keying (BPSK) or Quadrature Phase Shift Keying (QPSK) data stream with 40-100 percent of excess Bandwidth. This author suggests using two samples per symbol, and one of the two samples is used for the symbol decision. This author also uses the Wave Difference Method, but the only difference is that this author's approach does not require interpolation. Yazgan and Cavdar

[13] demonstrate that this in turn increases the Bit Error Rate (BER) and decrease the Signal to Noise Ratio (SNR).

The following figure shows the block diagram of the receiving modem that is proposed by Gardner [1]. The received data is divided into two streams which are the In-phase stream and the Quadrature stream. Then, the two streams are demodulated to convert the passband data to baseband data by two of quadrature-driven mixers. The carrier recovery branch is omitted from the block diagram because it is not related to the timing recovery algorithm. After the mixers, two data filters are used to avoid unwanted mixers products, remove the noise, and to shape the received data.

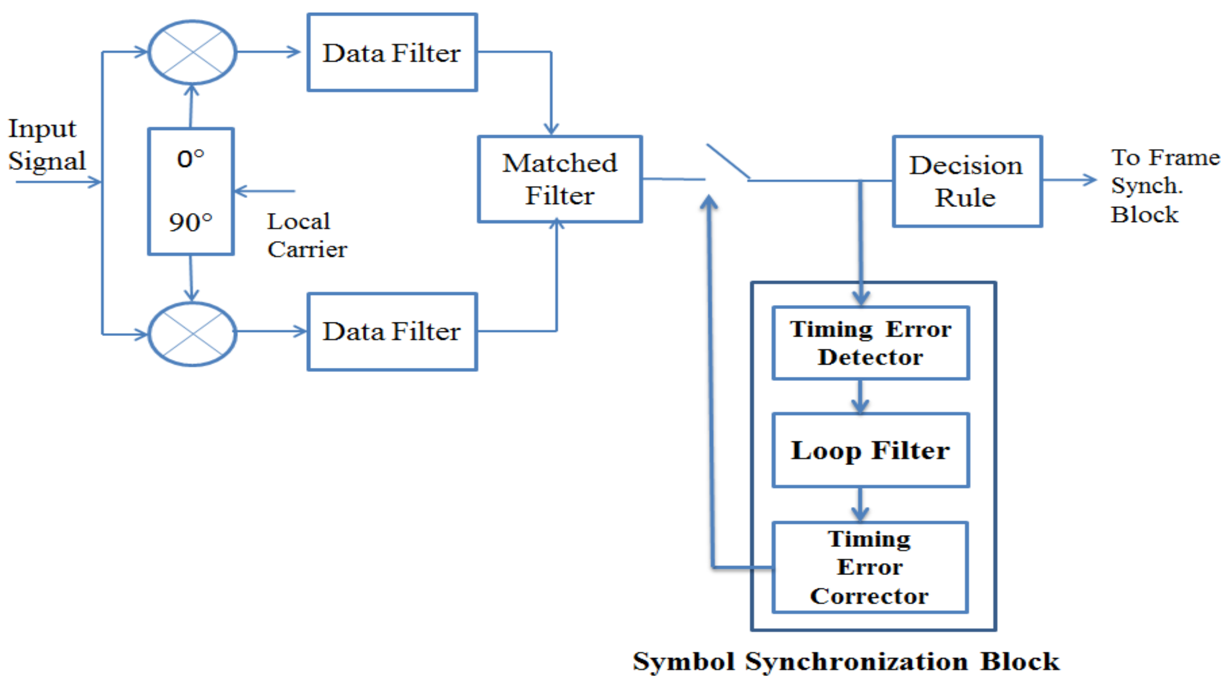

Figure 2.6: Typical modem (block diagram of Gardner) [1]

The output of the data filters consists of two real sequences $\{y_I(\ )\}$ and $\{y_Q(\ )\}$. Each symbol in these two sequences have two samples, and their timing information is used by the timing error detector. Gardner [1] states that, “One of the two samples occurs at the data strobe time, and the other one occurs midway between the data strobe times". Also, samples in these two sequences are transmitted, spaced by time interval *T.* The following error detector algorithm that is introduced by Gardner [1] can be showed by:

$$U_t(r)=y_I(r-½)[y_I(r)-y_I(r-1)] + y_Q(r-½)[y_Q(r)-y_Q(r-1)].$$

where r is the index that is used to designate the symbol number. The algorithm contains two parts, In-phase and Quadrature. In the In-phase part, the $y_I(r-½)$ is midway sample between the $y_I(r)$ sample and $y_I(r-1)$ sample. In the Quadrature part, the $y_Q(r-½)$ is midway sample between the $y_Q(r)$ sample and $y_Q(r-1)$ sample. The detector depends on the samples to find one error sample $U_t(r)$ for each symbol. The loop filter is used to smooth the error sequence, and then this error sequence is used to adjust a time error corrector.

Gardner's paper [1] concerns about the error detector, and it does not treat the loop filter or the error corrector. Gardner [1] proves that his algorithm (the Gardner algorithm) is independent of carrier phase. If Binary Phase Shift Keying (BPSK) modulation scheme is used in the communication system and in *I* channel, then the first part $\{y_I(\ )\}$ of the above formula will be used to find the timing information. Of course, the $\{y_Q(\ )\}$ will produce noise without any timing information. On the other hand, if Quadrature Phase Shift Keying (QPSK) modulation scheme is used in the communication system, then the both parts $\{y_I(\ )\}$ and $\{y_Q(\ )\}$ will be employed to find the timing information.

The Gardner algorithm [1] can be explained by finding the strobe samples on the both sides of the midway and in both the *I* and *Q* channels. The algorithm supposes that there is timing information when there is a transition between symbols, and the average midway sample should be zero when there is no timing error. When there is a timing error, the average of the midway sample gives nonzero magnitude, and the magnitude of this midway sample depends on the timing error value [1]. When there is no transition, the values of the strobe samples should be the same, and the difference between them gives zero. This means that there is no timing-error information when there is no transition, and in this case the value of the midway sample rejects.

Gardner [1] suggests using just the sign of the strobe samples to recover the symbols instead of the actual values. This, in Gardner's opinion, eliminates the noise effect if the data is filtering before taking the strobe samples. In this case, the signs of strobe samples are the optimum hard decision way to find symbols, and this makes the algorithm as a decision directed algorithm. This decision directed algorithm is similar to the digital transition tracking loop of Lindsey and Simon [14]. This thesis treats the loop filter and introduces five techniques to improve the communication system performance with taking into consideration the effect of the actual values of strobe samples.

2.4 Alamouti space-time code technique

The next generation systems require improving the quality of services such as quality of the voice or video. This improving increases the burden on the bandwidth and increase the Bit Error Rate (BER). In wireless telecommunication, radio signals reach to the receive antennas by two or more paths due to reflections and refractions from objects in the path of the signal. As mentioned by Alamouti [2, p1], multipath causes a destructive interference which results in fading called Rayleigh fading. In multipath fading environment, improving the BER from (10^-2) to (10^-3) requires improving the Signal to Noise Ratio by (10 dB). This improvement in SNR can be accomplished by reducing effects of Rayleigh fading without need to increase the power or to increase the burden on the bandwidth [2].

Alamouti [2, p.7] suggests using a technique to improve the diversity at all receivers in wireless systems. This can be achieved by using two transmitters and M receivers which will result in providing a diversity order of 2M. Consequently, this diversity improvement will positively affect Rayleigh fading. This technique does not require any feedback from the received antennas to transmitted antennas which is important for other techniques. Alamouti [2] shows that,"The scheme requires no bandwidth expansion, as redundancy is applied in space across multiple antennas, not in time or frequency."

Also, Alamouti [2] proved that the assumed diversity scheme reduces the error rates, and increases the capacity of wireless communication systems. The Alamouti technique can be used by all applications that are limited by the Rayleigh fading. The diversity is usually improved by using a one transmitter and M receivers which result in providing a diversity order of M, and this is called receive diversity. However, the Alamouti provides transmit diversity by using two transmitter and M receivers which result in providing a diversity order of 2M.

Normally in wireless networks, there are a one base station and multiple receive antennas. This states that the cost of transmit diversity is less than the cost of receive diversity because the first diversity requires just a one more transmit antenna instead of duplicating the number of all receive antennas. As a result, the Alamouti space-time code technique introduces low-cost solutions. Therefore, this technique meets the demand of markets by avoiding a complete redesign of exist systems and by improving quality and efficiency of wireless systems. Thus, the Alamouti technique is an appropriate choice for next-generation wireless systems because it reduces fading at receive antennas by using multiple transmit antennas at the base station.

In this thesis, a flat fading Rayleigh multipath channel is applied. Quadrature Phase Shift Keying (QPSK) modulation scheme is used with two transmitted antennas and a one received antenna. This diversity scheme is called the Alamouti Space Time Block Coding (STBC), and it can be explained as follow:

1- Assume that the transmitted sequence is $\{X_1, X_2, X_3, \ldots, X_n\}$

2- The transmitted sequence groups into two groups.

3- In the first time slot, X_1 and X_2 are transmitted from the first and second antenna. In the second time slot, $-X_2^*$ and X_1^* are transmitted from the first and second antenna. In the third time slot, X_3 and X_4 are transmitted from the first and second antenna. In the fourth time slot, $-X_4^*$ and X_3^* are transmitted from the first and second antenna and so on.

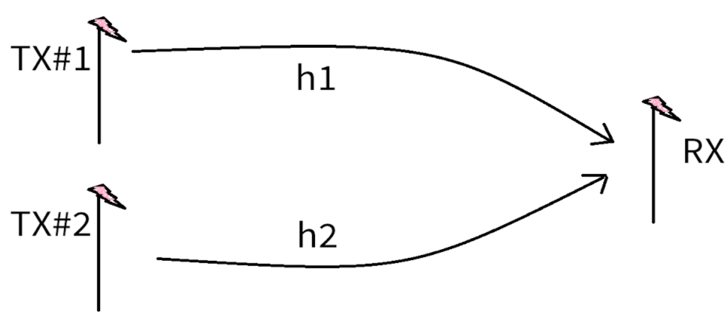

Figure 2.7: Alamouti scheme with two transmitted antenna and a one received antenna.

4- Each channel has different conditions. In other words, the Rayleigh fading in the first channel between the first transmitted antenna and the received antenna is different from the Rayleigh fading in the second channel between the second transmitted antenna and the received antenna. This means that each symbol is multiplied by h_i which is a complex number that is assigned randomly.

	Time t	Time t + T
Antenna #1	$x1$	$-x2^*$
Antenna #2	$x2$	$x1^*$

Table 2.2: The encoding and transmission sequence for the two branch transmit diversity scheme.

5- Although each channel is randomly varying, it is proposed that each channel remain constant over two time slots.

6- The noise on the received antenna in Gaussian distribution.

7- At the receive antenna, the channel h_i is assumed to be known.

8- The received signal at the first time slot is:

$$y_1 = h_1 \ x_1 + h_2 \ x_2 + n_1 \ .$$

And in the second time slot is:

$$y_2 = -h_1 \ x_2^* + h_2 \ x_1^* + n_2 \ .$$

where

y_1 and y_2 are the received symbols on the first and second time slot,

x_1 and x_2 are the transmitted symbols,

h_1 is the channel from first transmit antenna to receive antenna,

h_2 is the channel from second transmit antenna to receive antenna and

n_1 and n_2 are the noise in the first and second time slots respectively.

9- Then the above two signals go to the combiner that builds the following two signals

$$\hat{x}_1 = h_1^* \, y_1 + h_2 \, y_2^*.$$

$$\hat{x}_2 = h_2^* \, y_1 - h_1 \, y_2^*.$$

10- The above two signals are sent to the Maximum likelihood detector to find the final signal which compares with the original signal to find the Bit Error Rate (BER).

2.5 Gaps in the Literature

In the past studies, the researchers add new components to improve the quality of communication systems, but they ignored that these additional components increase the processing time in wireless communication systems. Also, the researchers use techniques that consume a part of the bandwidth as a feedback to synchronize between transmitters and receivers. Moreover, some researchers state that the bandwidth can be sacrificed or the power can be increased to manipulate the Rayleigh fading. Gardner [1] suggests using two samples per symbol for the timing recovery, but his technique consumes put burden on the bandwidth.

On the other hand, there are several techniques, such as the Alamouti technique, that can be used to eliminate from Rayleigh fading. However, the Alamouti technique requires working with receivers that are perfectly synchronized. Practically, it is hard to have receivers that are perfectly synchronized and do not use feedback data to estimate the channel condition in terms of the fading and the noise.

2.6 Theoretical perspective

This thesis depends on the estimation theory which provides the theoretical framework for studying the symbol timing problem. Also, this thesis presents ideas to improve the loop filter in the phase locked techniques depending on the Gardner technique. The Alamouti technique, in turn, improves the diversity order which reduces impacts of Rayleigh fading and increases the range of the coverage area. The previous studies state that the QPSK modulation scheme can be used to transmit high data rates. So, QPSK modulation scheme presents good solutions for increasing the demand on the bandwidth. As a result, this study will take advantage by using the Alamouti technique with timing synchronization technique that involves a new loop filter for QPSK modulation scheme.

Chapter 3

Methodology

3.1 Research Design

This research is a quantitative research because it involves an experimental design that can be accomplished by generating new algorithms by using Matlab. The results will state which one of the hypotheses will be realized. These results will also show conditions and requirements of wireless communication system. In other words, the simulation will present perspectives about the behavior of digital receivers with the new techniques in the wireless communication systems. Moreover, the techniques can be practically implemented in the future to demonstrate their limitations in wireless communication systems. Furthermore, the techniques, which are introduced to improve the performance of phase locked loops, take into consideration reducing the receiver complexities.

3.2 Elements of the experiment

Data streams, which are modulated by QPSK modulation scheme, will be used as input data for the wireless communication system. The procedure is a quasi-experiment because participants which are the data amount are not randomly assigned. In other words, there are several modulation schemes that can be used in this study such as Amplitude Modulation (AM), Frequency Modulation (FM), and Quadrature Amplitude Modulation (QAM), but only QPSK modulation scheme is used in this work.

3.3 Variables

3.3.1 Dependent Variables

The results of MATLAB are the dependent variables because these results depend on algorithms of symbol synchronization techniques. These results are represented by the Bit Error Rates (BER), the Signal to Noise Ratio (SNR), and convergence plots.

3.3.2 Independent Variable

The algorithms of symbol synchronization techniques are the independent variables because these algorithms affect the results of MATLAB. In other words, when these algorithms change, this means that the results also change.

3.4 Instrumentation and Materials

This thesis is a quantitative research that involves an experimental design. The instruments and materials that will be used in this study include:

- Synchronization algorithms.
- MATLAB program.
- Lab computer.
- Windows Operating System.
- Linux Operating System.

3.5 Procedure

Timing recovery algorithms are generated. These algorithms are simulated in MATLAB for symbol synchronization techniques. These algorithms represent a complete wireless communication system that involves a transmitter, an AWGN channel including Rayleigh fading, and a receiver. The number of filters and feedback loops in the wireless system are reduced as much as possible to decrease the processing time and reduce the complexities of the communication system. The modulation scheme is Quadrature Phase Shift Keying (QPSK), and the Alamouti technique is used as a diversity technique to eliminate the Rayleigh fading. Then, MATLAB program is run, and the results are appeared in plots. This allows assuring that symbol synchronization algorithms work properly and logically.

3.6 New techniques to improve the timing recovery in wireless receivers

In this section, five techniques are introduced to improve the timing recovery in wireless receivers. The main part in the wireless receiver is the Phase Locked Loop (PLL) which consists of three parts: the error detector, the loop filter, and the error corrector. The five techniques improve the performance of the timing recovery by depending on developing new algorithms for the loop filter. The error detector of the Gardner technique is used for the five techniques. The modulation schemes that are used are Binary Phase Shift Keying (BPSK) and Quadrature Phase Shift Keying (QPSK).

The baseline code of the Gardner technique is represented by a Matlab code, and convergence plots and plot SNR vs. BER are included. Each step of the baseline code is clarified, and characteristics of convergence plots are illustrated. Then, each technique that is used to improve the timing recovery performance is explained, and the results of each technique are compared to the baseline results. Then, the positive and negative characteristics are stated for each technique. These characteristics include the speed of the convergence, the Mean Squared Error

(MSE), SNR vs. BER behavior, and the level of complexities of the wireless receiver design that can result from each technique for both modulation schemes BPSK and QPSK.

3.7 Baseline Matlab code of the Gardner technique with BPSK

The first two sections in the code are the data generation and the pulse shape. The following code shows how the data amount is modulated:

Baseline_GardnerBPSK.m: Data generation and the pulse shape.

```
%% Data generation
N = 3*10^6;                                 % Number of bits
ip = rand(1,N)>0.5;                         % Generating 0,1
data = 2*ip-1;                              % BPSK modulation
Tsym = 100;                                 % No. of samples per symbol
MSE = zeros(1,5);                           % A memory for the MSE
BER_sim = zeros(1,5);                       % A memory for the simulated BER
%% Pulse shape
p = sin(2*pi*(0:Tsym-1)/(2*Tsym));          % Sinusoidal wave
data_up = zeros(1,length(data)*Tsym);% Creation a memory of zeros
data_up(1:Tsym:end) = data;                 % Interpolation the data
w = conv(data_up,p);                        % The convolution operation
```

The BPSK modulation scheme is used to transmit 3×10^6 symbols. `Tsym` states that one hundred samples per symbol are used in this code to simulate the impact of the interpolator. Then, pulse shape is applied on the signal. Each symbol is represented by the half of a sinusoidal cycle (this pulse shape is usually called a “half-sine”) that has one hundred samples.

After the pulse shape is done, the noise is added to the transmitted signal. The noise power is calculated so that the signal to noise ratio (SNR) varies from 2 dB to 10 dB. Then, the SNR values are converted into linear values. After that, a loop is created, and it is repeated for each SNR value to find values of noise that are added to the transmitted signal. After adding the noise, the detection and correction process starts at the receiver. This section starts with giving information about initial values.

According to the Gardner technique, the optimal value for midway samples should be at (n $\times$100), where n is the sequence of symbol in data. The first midway sample (called `center` in the code) is assumed to be received at 60. In addition to midway sample, the Gardner algorithm involves finding the "early sample" and the "late sample". The early and late samples can be calculated by finding the value of samples at (`center+delta`) and (`center-delta`). The value of delta is equal to the half symbol period which is equal to (`Tsym`/2=50).

After applying the Gardner algorithm, the shift value, that is used to correct the sampling operation, depends on the finding the average of a few iterations of the algorithm. In this code, the number of iterations is assumed to be equal to six (called `avgsamples` in the code). The Gardner technique supposes that the correction depends on the sign of the average result more than the value itself [1], so the `step size` is assumed to be equal to 1. Actually, this section of code is the focus of this thesis, so the development and the improvement are applied on it. After the initial values are given, another inside loop is created to conduct the error detection, loop filter, and the correction operations.

Next, the code creates a loop that depends on total number of samples in the received signal. The loop variable `ii` represents the index of the midway samples. The index must start after `delta` value to allow finding the first early sample. In the same meaning, the index must end before the last sample in the received signal by `delta` value to allow finding the last late sample. Then, the midway, late, and early samples are calculated. Either early samples or late samples can be used to recover the symbols of data. In this Matlab code, early samples are used to recover symbols of received signal. When the end of the received signal is reached, the loop ends.

Error detection is simulated by applying the subtraction process and the Gardner algorithm. Then, the loop filter is simulated by finding the mean of several instantaneous timing estimates. If the sign of the mean is positive, the `tau` value is equal to -1. Otherwise, the `tau` value is equal to 1. Next, the rest of the Matlab codes are completed. The remind step is used to state how fast the convergence happens. When the convergence happens, this means that samples are taken close or at the optimal value. In other words, the samples at these optimal times are less affected by the noise; consequently, there are fewer errors in the symbols' detection process.

In order to use the original convergence plots in the comparison with convergence plots of the five techniques, the Mean Squared Error can be used in this comparison. According to [15], the Mean Squared Error (MSE) can be calculated by applying the following formula:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (\hat{Y}_i - Y_i)^2.$$

where n: the number of iterations.

$\hat{Y}_i$: the estimated tau.

Y_i : the optimal tau.

Then to find the MSE, values of tau are saved in a vector. Next for the correction, the value of tau is added to the current center plus the 100 that is necessary to move to the next symbol. The following code demonstrates the above sections:

Baseline_GardnerBPSK.m: Noise addition,detection and correction.

```
%% Noise addition
SNRdB=2:2:10;                 % Signal to Noise Ratio
SNR=10.^(SNRdB/10);           % The linear values for the noise
for cv=1:length(SNR)          % Generate a loop
    noise=sqrt(1/(2*SNR(cv)))*randn(1,length(w));
                              % Noise  generation
```

```
received=w+noise;       % Received signal with noise
%% Detection and Correction
tau=0;                  %Initial value for tau
delta=Tsym/2;           %The shifting value before and
                        %after the midway sample
center=60;              %The assumed place for the first
                        %midway sample
a=zeros(1,N-1);         %A memory of zeros for the
                        %earlier samples
cenpoint=zeros(1,N-1);  %A memory of zeros for the
                        %midway samples
remind=zeros(1,N-1);    %A memory of zeros for the remind
avgsamples=6;           %Six values of Gardner algorithm
                        %are used to find the average
stepsize=1;             %Correction step size
rit=0;                  %Iteration counter
GA=zeros(1,avgsamples); %A memory of zeros
tauvector=zeros(1,1900);%A memory of zeros for tau
                        %vector(2000-100=1900)
uor=0;                  %A counter for the tau vector

for ii= (Tsym/2)+1:Tsym:length(received)-(Tsym/2)
    rit=rit+1;                          %A counter
    midsample=received(center);         %The midway sample
    latesample=received(center+delta);%The late sample
    earlysample=received(center-delta);%The early sample
    a(rit)=earlysample;                 %Save samples
    %% Error detection
    sub=latesample-earlysample;         %Subtraction process
    GA(mod(rit,avgsamples)+1)=sub*midsample;
                                        %Gardner Algorithm
    %% Loop filter
    if mean(GA) > 0
       tau = -stepsize;                 %Shift by decreasing
    else
      tau = stepsize;                   %Shift by increasing
    end
    %% Safe remind values
    cenpoint(rit)=center;               %Save positions of
                                        %midway samples
    remind(rit)=rem((center-Tsym/2),Tsym);
                                        %Save remind values to find
                                        % convergence plots
   %% tau vector
   if rit>=100 && rit<2000              %tau vector from 100 to
```

```
                                        %2000 where the convergence
        uor=uor+1;                      %happens
        tauvector(uor)= (remind(rit) - (Tsym/2)).^2;
                                        %Difference between the
                                        %estimated tau & the
     end                                %optimal tau
     %% Correction
     center=center+Tsym+tau;            %Adding the tau value
     if center>=length(received)-(Tsym/2)-1
        break;                          %Break the loop when
                                        %the midway sample reaches
                                        %to 51 samples before
                                        %the last sample
     end
  end
```

Now, it is necessary to state that the value of `tau` depends on error information which can be obtained when there is a transition between symbols. The next four scenarios shows how the `tau` value is taken.

Scenario #1

The first scenario supposes that there is a BPSK signal that contains just two bits [1, -1]. The following figure shows where the early, midway, and late samples are taken:

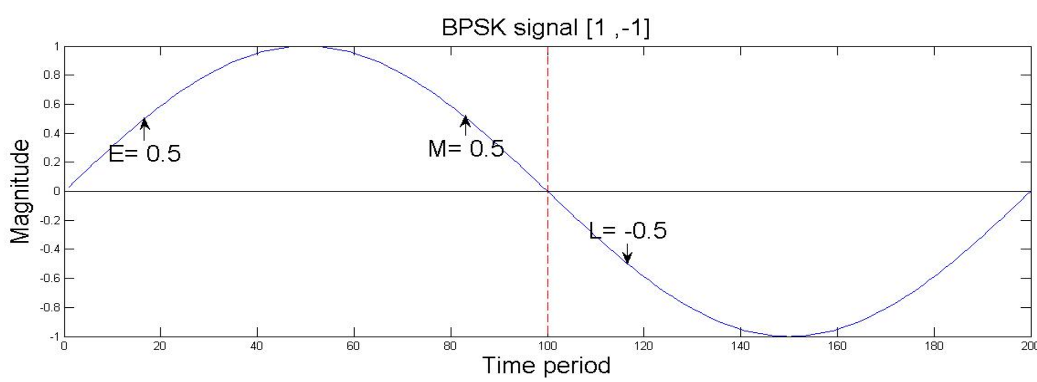

Figure 3.1: shows the early, midway, and late samples of the first scenario

By using the Gardner algorithm, the result will be as below:

$GA = M * (L - E)$

$= 0.5 * (-0.5 - 0.5) = -0.5$

where GA is the Gardner algorithm.

M is the midway sample.

L is the late sample.

E is the early sample.

The optimal midway samples should be taken at ($n \times 100$) where n is the sequence of the symbol. The midway sample in figure 3.1 is before the optimal value, so it should be shifted forward by the `step size`. The Gardner [1] uses the sign of the mean of several instantaneous timing estimates and ignores the actual value of the mean in the correction operation. As a result, when the sign is minus, the `tau` should be 1.

Scenario #2

The second scenario supposes that there is a BPSK signal that also contains just two bits [1, -1]. The following figure shows where the early, midway, and late samples are taken:

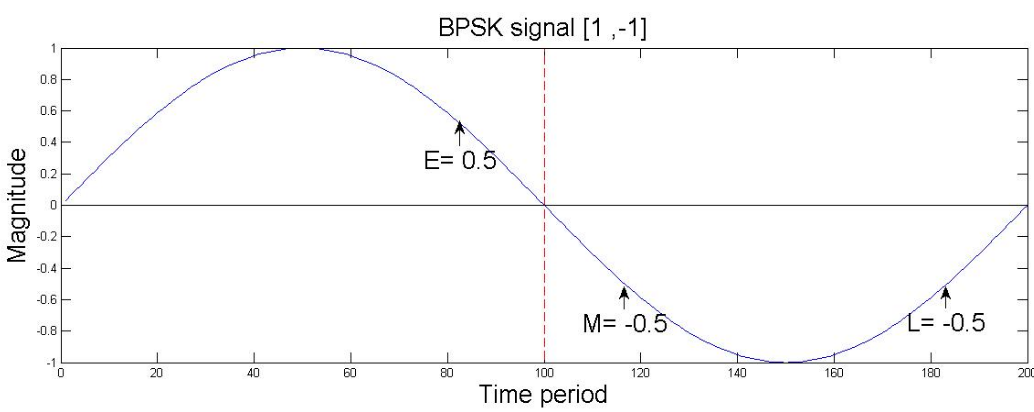

Figure 3.2: shows the early, midway, and late samples of the second scenario

By using the Gardner algorithm, the result will be as below:

$GA = -0.5 * (-0.5 - 0.5) = 0.5$

The optimal midway samples should be taken at $(n \times 100)$. The midway sample in figure 3.2 is after the optimal value, so it should be shifted backward by the `step size`. As a result, when the sign is positive, the `tau` should be -1.

Scenario #3

The third scenario supposes that there is a BPSK signal. This signal contains just two bits [-1, 1]. By using the Gardner algorithm on the samples of figure 3.3, the result will be as below:

$GA = -0.5 * (0.5 - (-0.5)) = -0.5$

The optimal midway samples should be taken at $(n \times 100)$. The midway sample in figure 3.3 is before the optimal value, so it should be shifted forward by the `step size`. As a result, when the sign is negative, the `tau` should be 1.

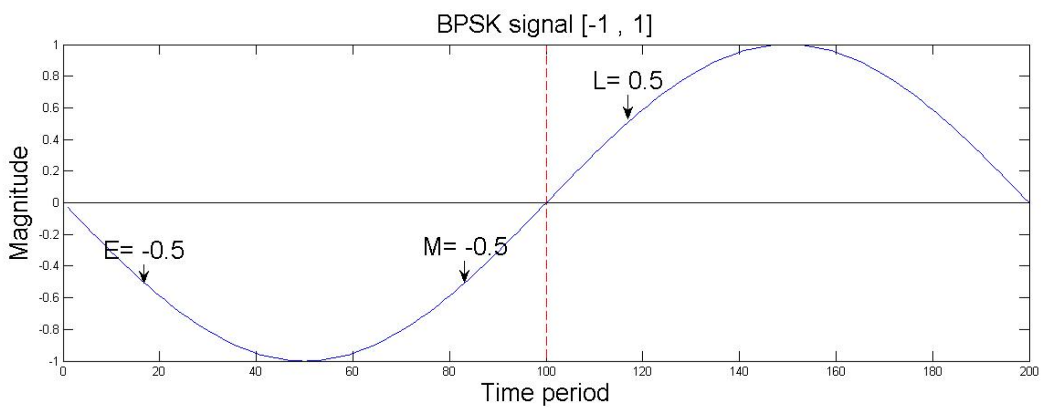

Figure 3.3: shows the early, midway, and late samples of the third scenario

Scenario #4

The fourth scenario supposes that there is a BPSK signal that also contains just two bits [-1, 1]. The following figure shows where the early, midway, and late samples are taken:

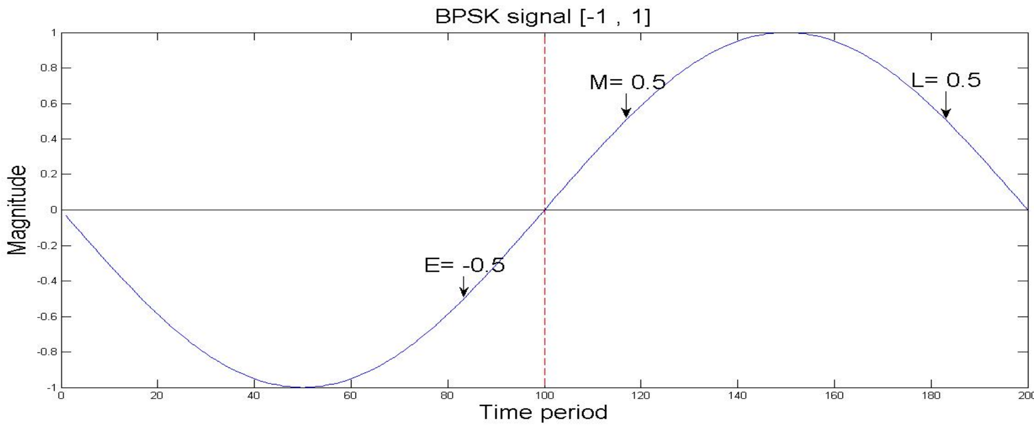

Figure 3.4: shows the early, midway, and late samples of the fourth scenario

By using the Gardner algorithm, the result will be as below:

$$GA = 0.5 * (0.5 - (-0.5)) = 0.5$$

The optimal midway samples should be taken at $(n \times 100)$. The midway sample in figure 3.4 is after the optimal value, so it should be shifted backward by the `step size`. As a result, when the sign is positive, the `tau` should be -1.

The above four scenarios states that `tau` should be 1 when the sign of the mean of several instantaneous timing estimates is minus. However, the `tau` should be -1 when the sign of the mean of several instantaneous timing estimates is positive.

After this, the mean is taken to find the value of MSE for the tau vector after the end of the loop of a certain SNR. Next, the remind values are plotted to show the convergence plots. The next section in the Matlab code is the Bit Error Rates (BER) calculation. First, the total error is computed as it is stated in the following code. When the data is equal to 1 and the received sample is less than zero, this considers as an error. Similarly, when the data is equal to -1 and the received sample is more than zero, this also consider as an error. Second, the BER is calculated by dividing the total computed error over the all data amount. The loop of the SNR values ends after computing the BER. The following code demonstrates MSE, convergence plot, and the BER calculation sections:

Baseline_GardnerBPSK.m: MSE, convergence plot, and the BER calculation.

```
    %% Mean Squared Error (MSE)
    MSE(cv)=mean(tauvector);          %Finding the Mean Squared Error
    %% convergence plot
    figure
    symbols = 200;
    subplot(2,1,1);
    plot(remind(1:symbols),'*-');
    hold on
    lim1=40*ones(1,symbols);
    lim2=60*ones(1,symbols);
    plot(lim1);
    hold on
    plot(lim2);
    title('Convergence plot for BPSK-Gardner');
    ylabel('tau axis'), xlabel('iterations')
    legend( ['SNRdB=' int2str(SNRdB(cv))]);
    axis([1 symbols 0 Tsym]);
    subplot(2,1,2);
    symbols = 2000;
    plot(remind(1:symbols),'*-');
    hold on
    plot(lim1);
    hold on
    plot(lim2);
    title('Convergence plot for BPSK-Gardner');
    ylabel('tau axis'), xlabel('iterations')
    legend( ['SNRdB=' int2str(SNRdB(cv))]);
    axis([1 symbols 0 Tsym]);
    %% Calculating the simulated BER
    Error=0;                         %Set the initial value for Error
    for k=1:N-1                      %Error calculation
        if ((a(k)>0 && data(k)==-1)||(a(k)< 0 && data(k)==1))
               Error=Error+1;
       end
    end
    BER_sim(cv)=Error/(N-1);         %Calculate error/bit
end
```

Finally, the theoretical BER is calculated as it is illustrated in the next code. Then, the SNR vs. BER figure is plotted as it is showed in the following code:

Baseline_GardnerBPSK.m: SNR vs BER plot.

```
%% Plot SNR Vs BER
BER_th=(1/2)*erfc(sqrt(2*SNR)/sqrt(2));          %Calculate The
                                                  %theoretical BER
figure
semilogy(SNRdB,BER_th,'k-','LineWidth',2);       %Plot theoretical BER
hold on
semilogy(SNRdB,BER_sim,'r-','LineWidth',2);      %Plot simulated BER
title('SNR Vs. BER for BPSK-Gardner technique');
legend('Theoretical','Simulation');
ylabel('log BER');
xlabel('SNR in dB');
```

The above Matlab codes represent the baseline codes of the Gardner algorithm. Note that the initial value for the midway sample is taken at 60, so the first early sample is taken at:

First early sample = Initial value of the midway sample - `delta`= 60 - 50 = 10.The goal of the above codes is to take the midway samples at or close to ($n \times 100$). This means that the early samples, which are used to recover the data, are taken at or close to ($n \times 50$). The following figures states the original convergence plots for BPSK-Gardner technique with BER plot.

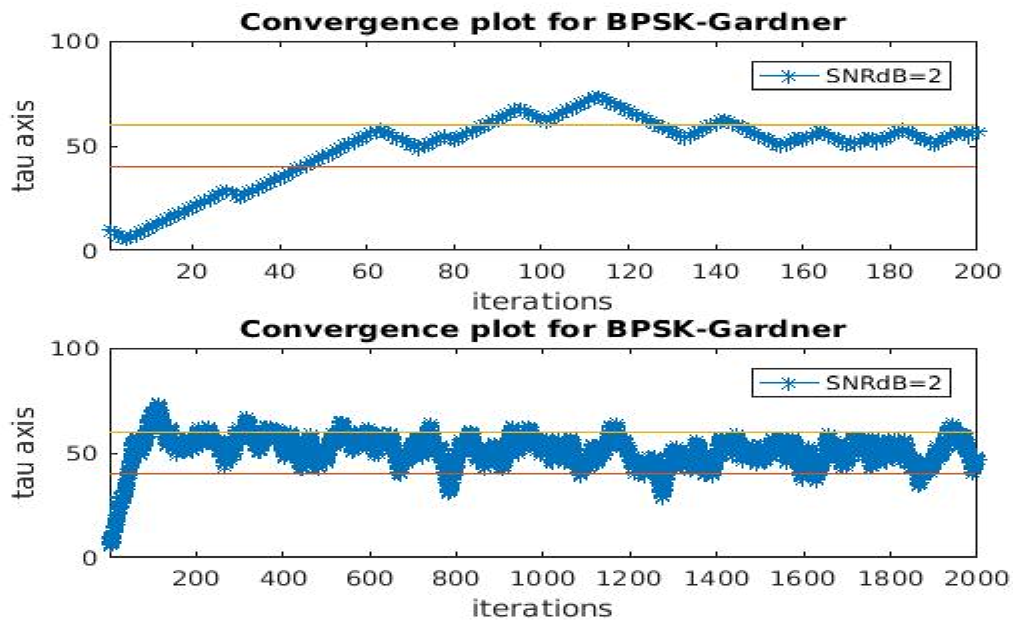

Figure 3.5: BPSK-Gardner technique, SNR= 2 dB

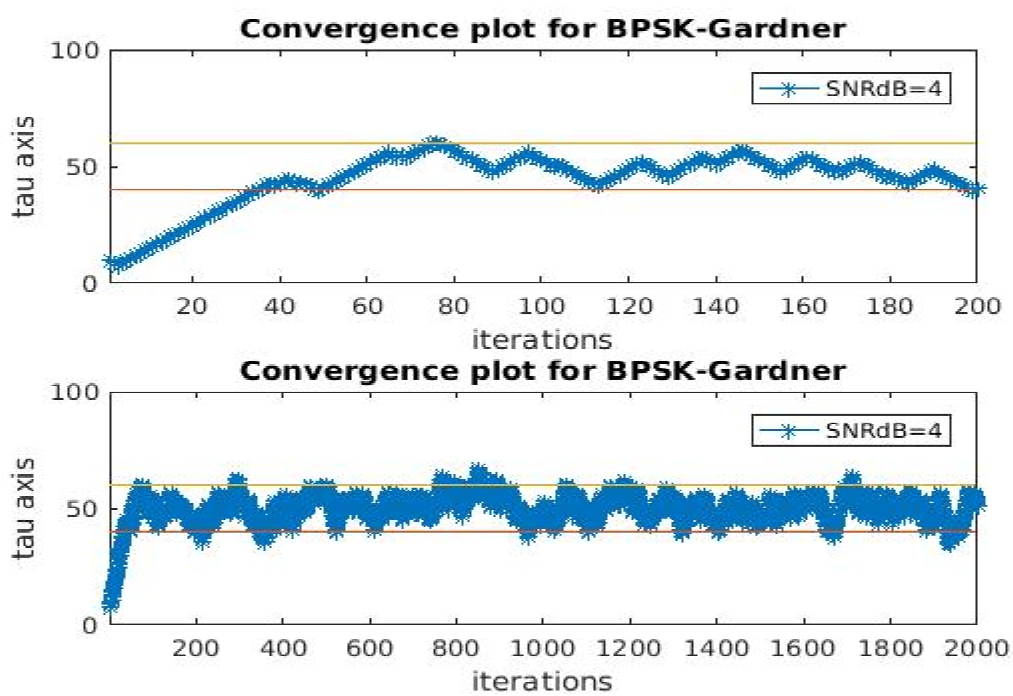

Figure 3.6: BPSK-Gardner technique, SNR=4 dB

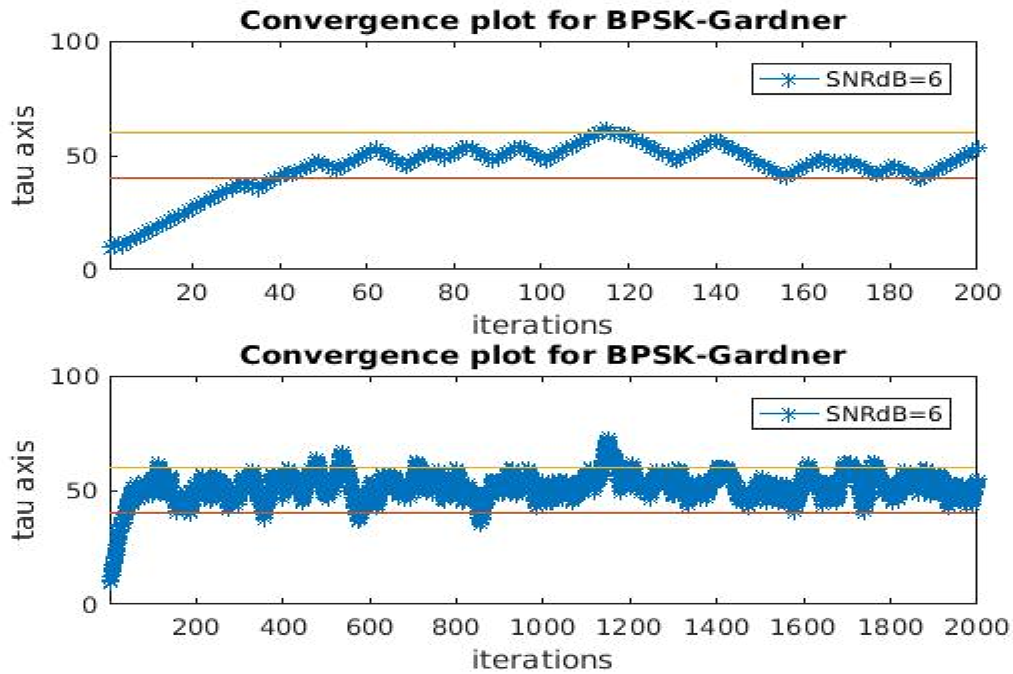

Figure 3.7: BPSK-Gardner technique, SNR= 6 dB

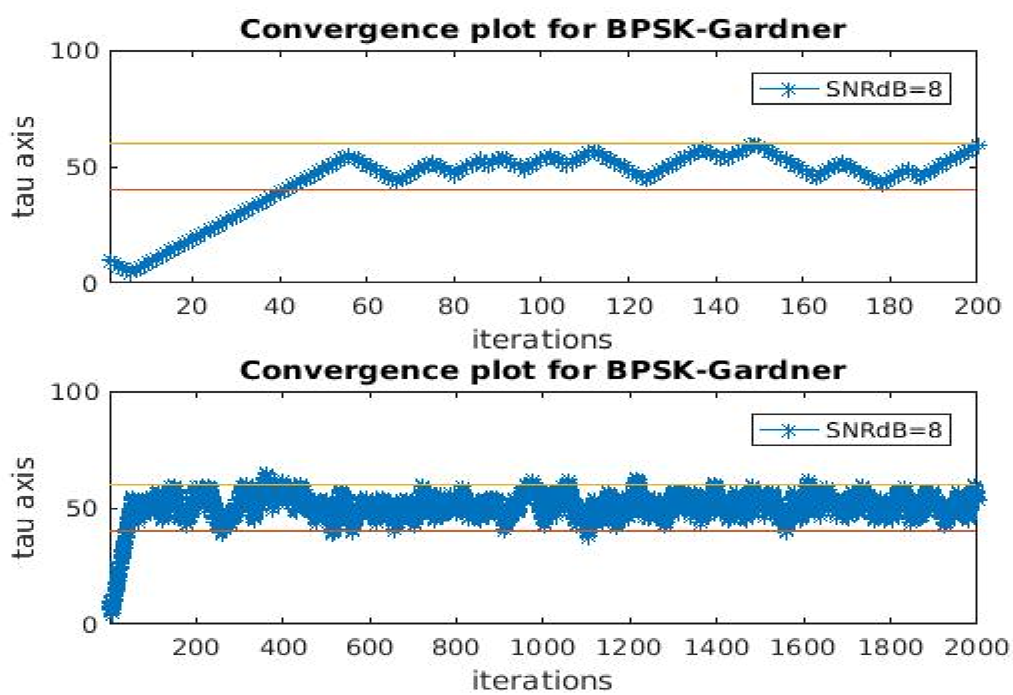

Figure 3.8: BPSK-Gardner technique, SNR= 8 dB

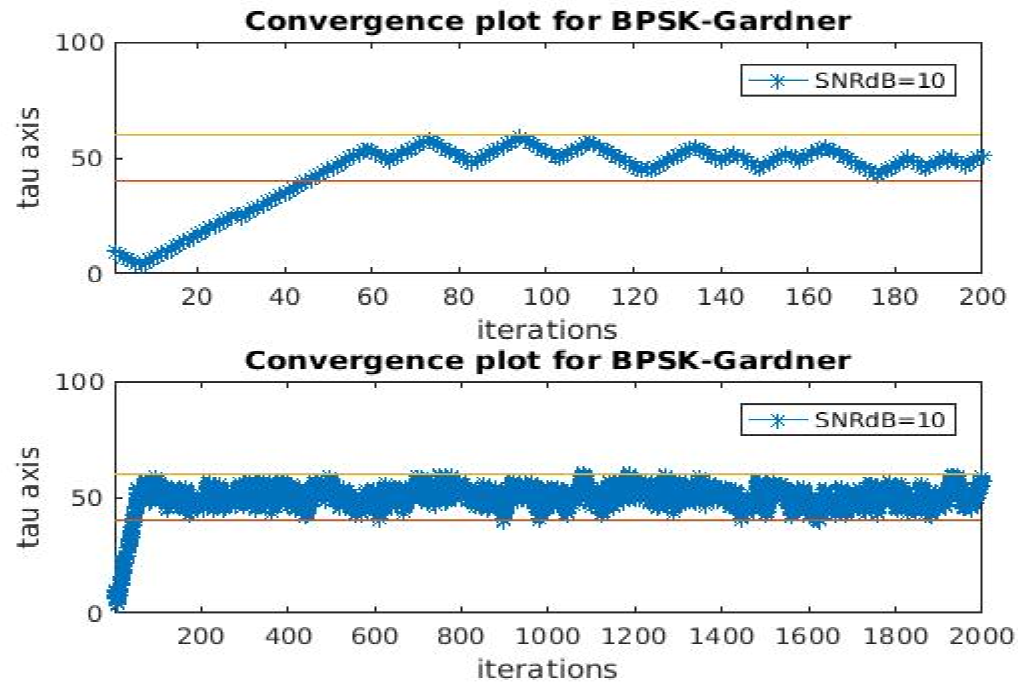

Figure 3.9: BPSK-Gardner technique, SNR=10 dB

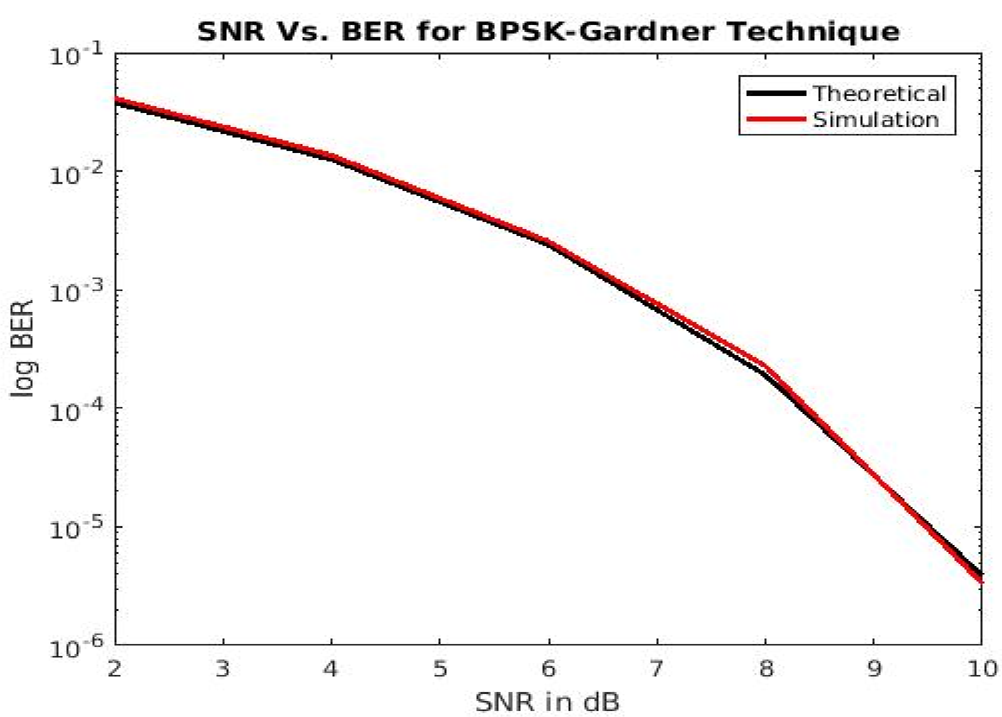

Figure 3.10: BPSK-Gardner technique, SNR vs. BER plot

Finally, the convergence plots can be used in the comparison operation with convergence plots of the five techniques. The following table states the values of MSE that are corresponding to the SNR values for the original convergence plots.

SNR	2 dB	4 dB	6 dB	8 dB	10 dB
MSE	36.9274	28.8579	23.5400	18.2811	13.9632

Table 3.1: states SNR values with MSE value- Gardner technique

3.8 The first technique to improve the timing recovery

This technique works on the error detection code and on the loop filter code. Otherwise, all other sections of the baseline Matlab code are used in this technique. This technique introduces a factor equal to -20 that can be multiplied by the result of error samples for the Gardner algorithm to give a faster correction operation. As it is mentioned before, this thesis assumes that there are one hundred samples per each symbol to simulate the effect of the interpolator. This means that there are one hundred magnitudes in each symbol. The maximum magnitude for the BPSK symbol is equal to 1 in the positive part, and the magnitude is equal to -1 in the negative part. In other words, the peak to peak magnitude for each symbol is equal to 2. As a result, there are one hundred different magnitudes in this range from 1 to -1. So, the difference in the magnitude between a one sample and the next sample is equal to 0.02. The following equation states that:

$$D = \frac{V\{p-p\}}{Tsym} = \frac{2}{100} = 0.02$$

where

D: is the difference in the magnitude between one sample and the next sample.

$V\{p-p\}$: is the magnitude from peak to peak which is equal to (2).

Tsym: is the number of samples per symbol.

One of things in this technique that should be understood is that the value of *D* changes when the number of samples per symbol changes. To understand this technique, the following scenario introduces a good explanation that starts from the end to find the factor. This scenario supposes that the transmitted data are [1, 0], so this means that the BPSK is [1, -1]. After the pulse shape, the output is a sine wave signal. The following figure represents these two bits:

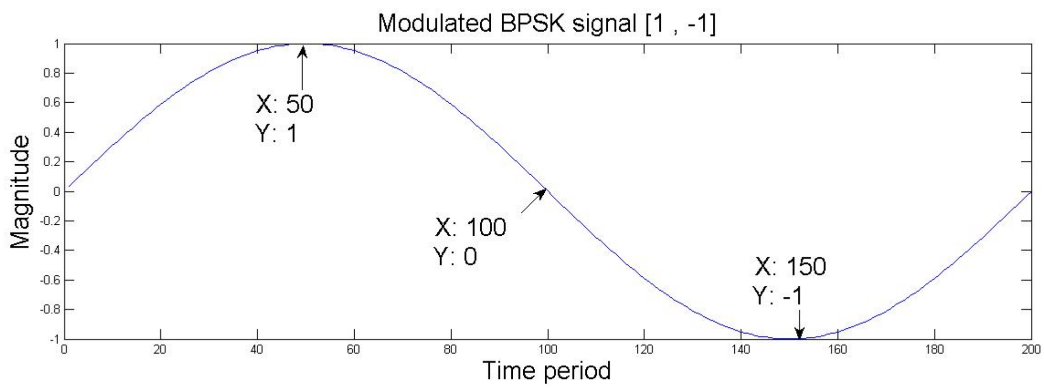

Figure 3.11: BPSK signal represents [1, -1]

At optimal time, the first midway sample should be received at sample#100, and its magnitude should be equal to zero when there is no noise. The earlier sample should be equal to (1) at sample#50, and the late sample should be equal to (-1) at Sample# 150.

The following scenario#1 supposes that the first midway sample is received at 101. Theoretically, this means that the midway sample is one step away from the optimal position and in the negative part, and its magnitude is

$$Midway\ sample = Nu \times D\,.$$

$$= 1 \times 0.02 = -0.02.$$

(The minus sign is added because the sample is received in negative part) where

Nu: is the number of steps that is supposed to be taken for the correction operation. The following figure shows that:

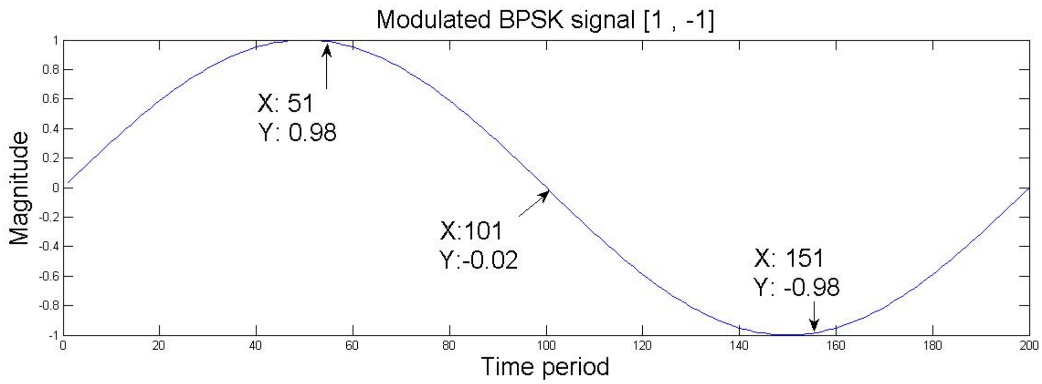

Figure 3.12: BPSK signal represents [1, -1]

The earlier sample is also one step away from the maximum magnitude which is (1), so the magnitude of the earlier samples should be:

$$Earlier\ sample = 1 - (Nu \times D).$$

$$= 1 - (0.02) = 0.98$$

Similarly, the late sample is also one step away from the minimum magnitude which is (-1), so the magnitude of the late samples should be:

$$Earlier\ sample = -1 + (Nu \times D).$$

$$= -1 + (0.02) = -0.98$$

By applying the Gardner algorithm (GA) which is:

$$GA = (late\ sample\ -\ earlier\ sample) \times\ midway\ sample$$

$$GA = (-0.98 - 0.98) \times\ 0.02 = 0.0392$$

The value (0.0392) should be used in the loop filter to make a one backward step. The logic way to understand that (0.0392) is equal to one backward step is by taking the round after multiplying it by the factor (− 20):

$$tau = round\,(GA \times -20).$$

$$= round\,(0.0392 \times -20)$$

$$= round\,(-0.784) = -1$$

The value of this factor (-20) changes when the value of *D* changes. As mentioned previously, *D* changes when the number of samples per symbol changes. The following formula gives the right factor depending on the number of samples per symbol:

$$Factor = \frac{V(p-p)}{Tsym} \times \frac{-(Tsym)^2}{10} = D \times \frac{-(Tsym)^2}{10}.$$

When $V(p-p)$=2 (peak to peak BPSK magnitude) and Tsym=100 (The number of samples per symbol), the factor is:

$$Factor = \frac{2}{100} \times \frac{-(100)^2}{10} = -20$$

When $V(p-p)$=2 (peak to peak BPSK magnitude) and Tsym=1000 (The number of samples per symbol), the factor becomes:

$$Factor = \frac{2}{1000} \times \frac{-(1000)^2}{10} = -200$$

Now, what if the midway sample is received in a different position? Does the proposed technique have the ability to conduct the correction operation? The questions can be answered by the following scenario#2 which supposes that the first midway sample is received at 103. This means that the midway sample is 3 steps away from the optimal position and in the negative part, and its magnitude is

$$Midway\ sample = Nu \times D$$

$$= 3 \times 0.02 = -0.06$$

(The minus sign is added because the sample is received in negative part)

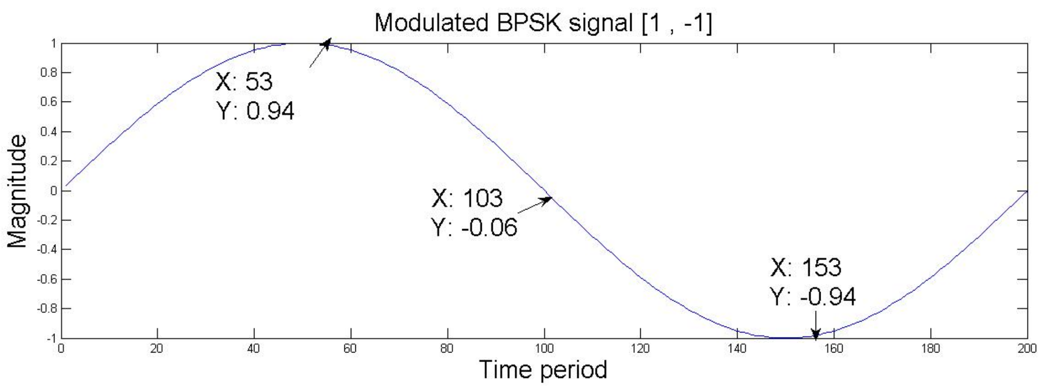

Figure 3.13: BPSK signal represents [1, -1]

The earlier sample is also 3 steps away from the maximum magnitude which is (1), so the magnitude of the earlier samples should be:

$$Earlier\ sample = 1 - (Nu \times D)$$

$$= 1 - (0.06) = 0.94$$

Similarly, the late sample is also 3 steps away from the minimum magnitude which is (-1), so the magnitude of the late samples should be:

$$Earlier\ sample = -1 + (Nu \times D) = -1 + (0.06) = -0.94$$

By applying the Gardner algorithm (GA) which is:

$$GA = (late\ sample - earlier\ sample) \times midway\ sample$$

$$GA = (-0.94 - 0.94) \times 0.06 = 0.1128$$

The value (0.01128) should be used in the loop filter to make 3 backward step. By using the same factor which is (-20), tau will be:

$$tau = round\ (GA \times -20) = round\ (0.1128 \times -20) = round\ (-2.256) = -2$$

Now, the two backward steps make the next midway sample happens at 201, and this can be shown by:

$$New\ midway\ sample = previous\ midway\ sample + Tsym + tau$$

$$= 103 + 100 + (-2) = 201$$

At 201, the magnitude of the midway sample has exactly the same magnitude at the 101 when there is no noise. So, by applying the scenario#1, this make the next tau equal to (-1). As a result, the tau becomes equal to (-3) after the loop run twice, and this tau results from combining the two scenarios. In other words, tau = -2 when the midway sample is taken at 103, and tau = -1 when the midway tau is taken at 201. In addition to that, a weighted filter is used in this technique. The next Matlab code illustrates using the factor and the weighted filter:

first_technique.m: Error detection and loop filter.

```
%% Error detection
sub=latesample-earlysample;    %Subtraction process
GA=sub*midsample;              %Gardner Algorithm
%% Loop filter
tau=round(GA*-20);             %Using the factor (-20)
if tau>4                       %tau= 4 when tau is larger than(4)
     tau=4;
elseif tau<-4                  %tau= -4 when tau is smaller than -4
      tau=-4;
end
```

The convergence plots and SNR vs. BER plot of the first technique are shown below:

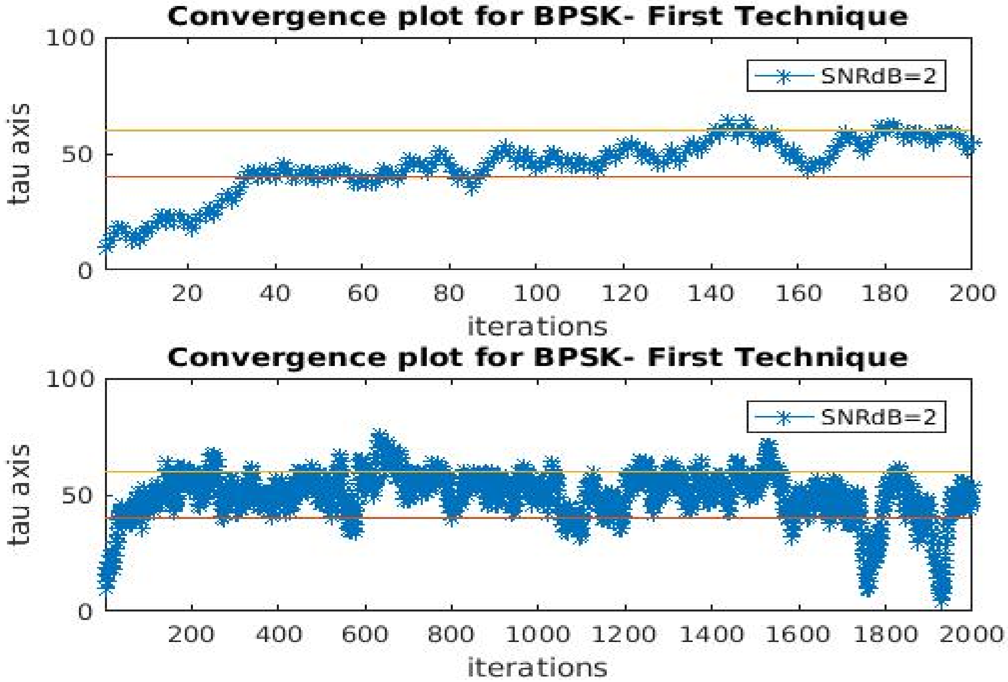

Figure 3.14: BPSK-first technique, SNR=2 dB

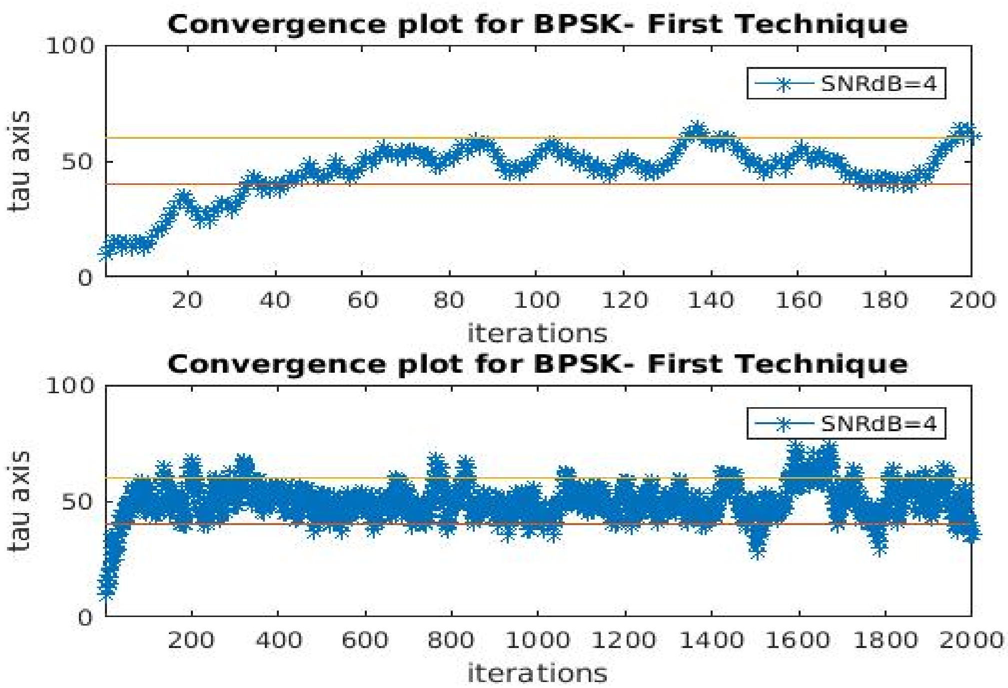

Figure 3.15: BPSK-first technique, SNR=4 dB

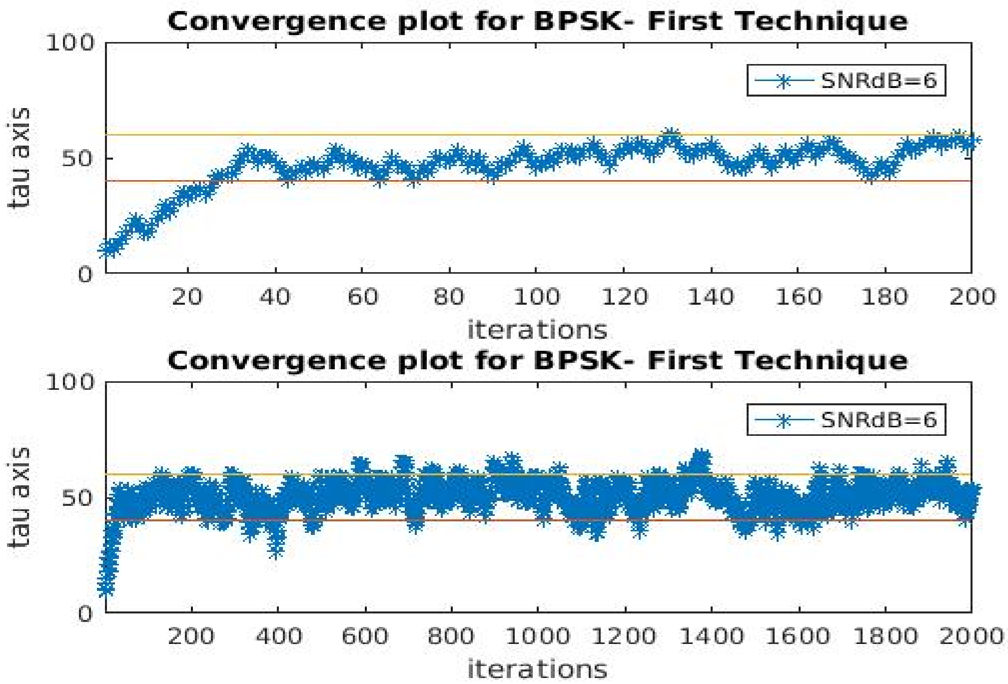

Figure 3.16: BPSK-first technique, SNR=6 dB

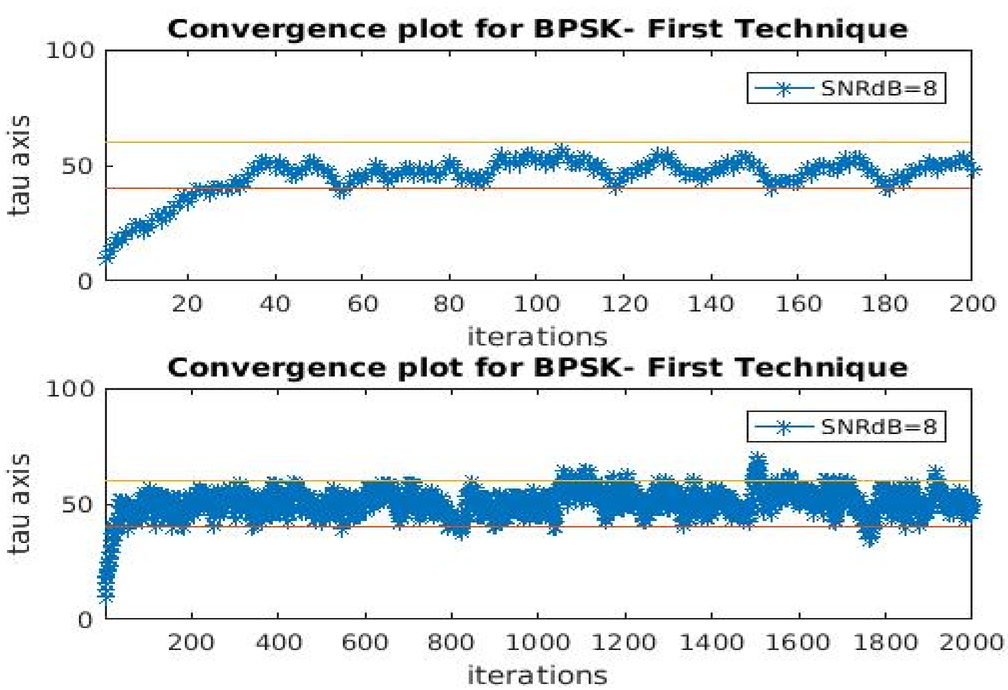

Figure 3.17: BPSK-first technique, SNR=8 dB

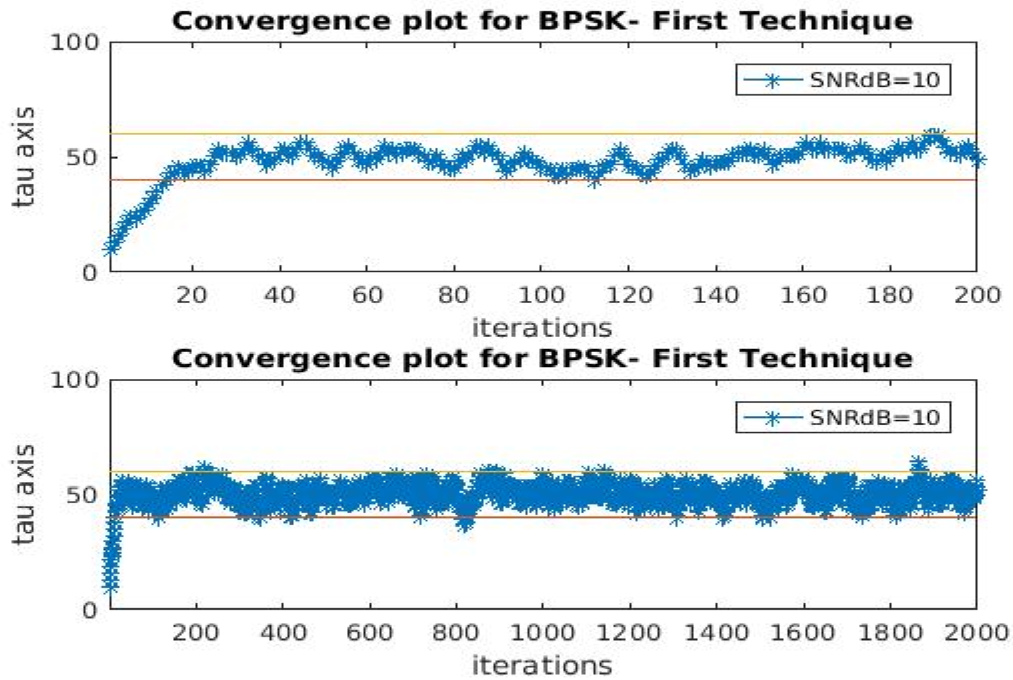

Figure 3.18: BPSK-first technique, SNR=10 dB

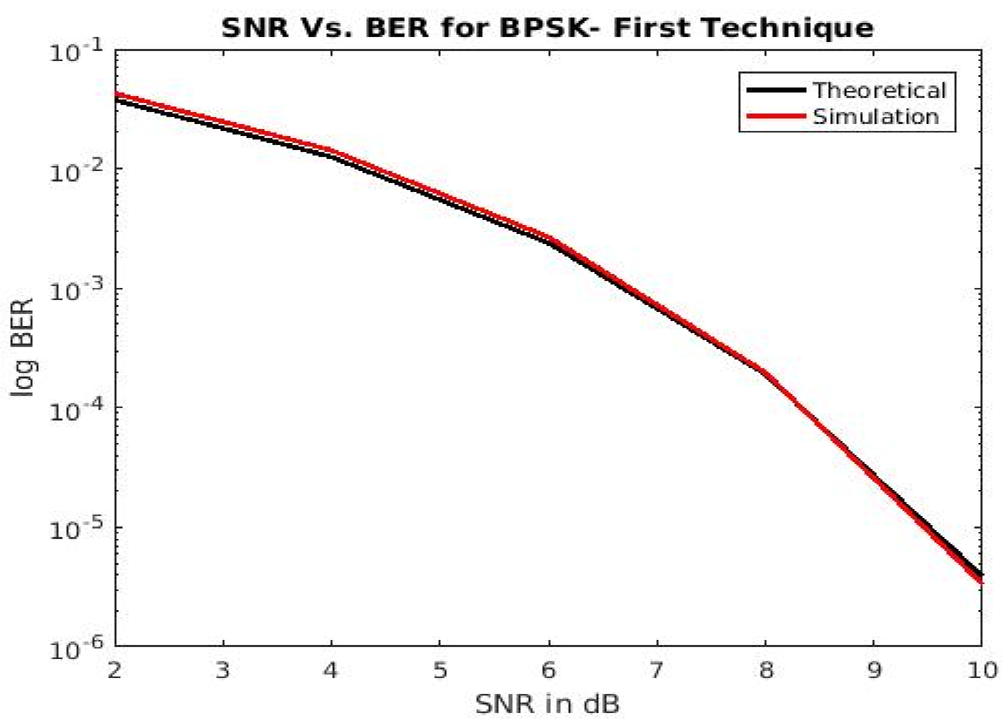

Figure 3.19: BPSK- first technique, SNR vs. BER plot

The next table reveals the MSE values that are corresponding to the SNR values for the first technique.

SNR	2 dB	4 dB	6 dB	8 dB	10 dB
MSE	54.3821	49.5479	31.5684	24.3621	16.7868

Table 3.2: states MSE values with SNR values- first technique

3.8.1 Evaluation of the first technique

To evaluate the first technique, it is important to compare the results of this technique with the results of the Gardner technique. In term of the convergence plot, the first technique has a faster convergence comparing to the Gardner technique. In term of the SNR vs. BER plot, both the first technique and the Gardner technique have the same SNR vs. BER behavior (see figures 3.10 and 3.19). Although the first technique has higher values of MSE (see table 3.2), the BER results of the first technique still have the same BER results of the Gardner technique. In term of the complexity, the first technique eliminates the average filter that is used by the Gardner technique, so the first technique reduces the complexity of the digital receiver structure, and reduces the processing time; as a result, the first technique improves the performance of the communication system.

3.9 The second technique to improve the timing recovery

This technique has the same baseline code except the loop filter code. In the baseline code of the Gardner algorithm, error time information depends on the sign of the mean of the Gardner algorithm. In this technique, the actual value of the mean is used to indicate how many step sizes should be taken to conduct the correction operation. To do that, a weighted filter should be used as it is demonstrated in the following Matlab code:

second_technique.m: Loop filter.

```
%% Loop filter
     if mean(GA)>0 && mean(GA)<0.3
        tau=-1;                              %Shift is decreasing by 1
    elseif mean(GA)>0.3 && mean(GA)<0.6
        tau=-2;                              %Shift is decreasing by 2
    elseif mean(GA)>0.6
        tau=-3;                              %Shift is decreasing by 3
    elseif mean(GA)<0 && mean(GA)>-0.3
        tau=1;                               %Shift is increasing by 1
    elseif mean(GA)<-0.3 && mean(GA)>-0.6
        tau=2;                               %Shift is increasing by 2
    elseif mean(GA)<-0.6
        tau=3;                               %Shift is increasing by 3
    else
        tau=0;                               %There is no shift
    end
```

As it is shown in the above code, a weighted filter is used to find the value of the mean of the Gardner algorithm that gives seven possibilities for the step size which (1,2,3,-1,-2,-3, & 0) instead of two possibilities as it is used in the baseline code which are (1,-1). The value of the mean of the Gardner algorithm is high when the samples are taken far away from the optimal time. So, it is logically to take more than one step when the magnitude is high. The new loop filter should increase the convergence fast. The following figures demonstrate the convergence plots and the SNR vs. BER plot.

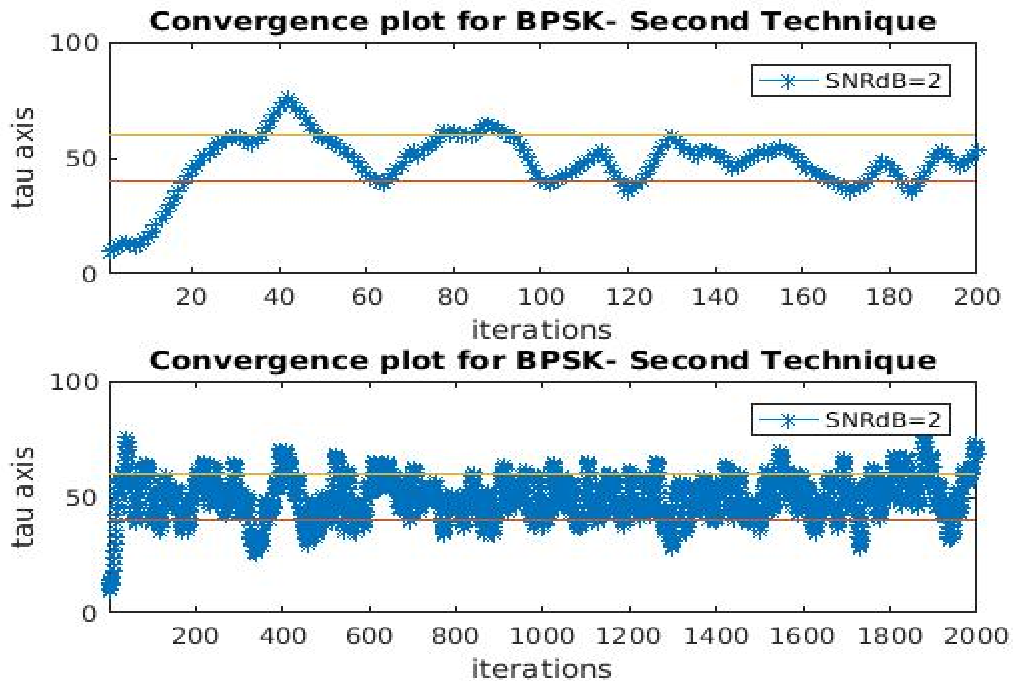

Figure 3.20: BPSK-second technique, SNR=2 dB

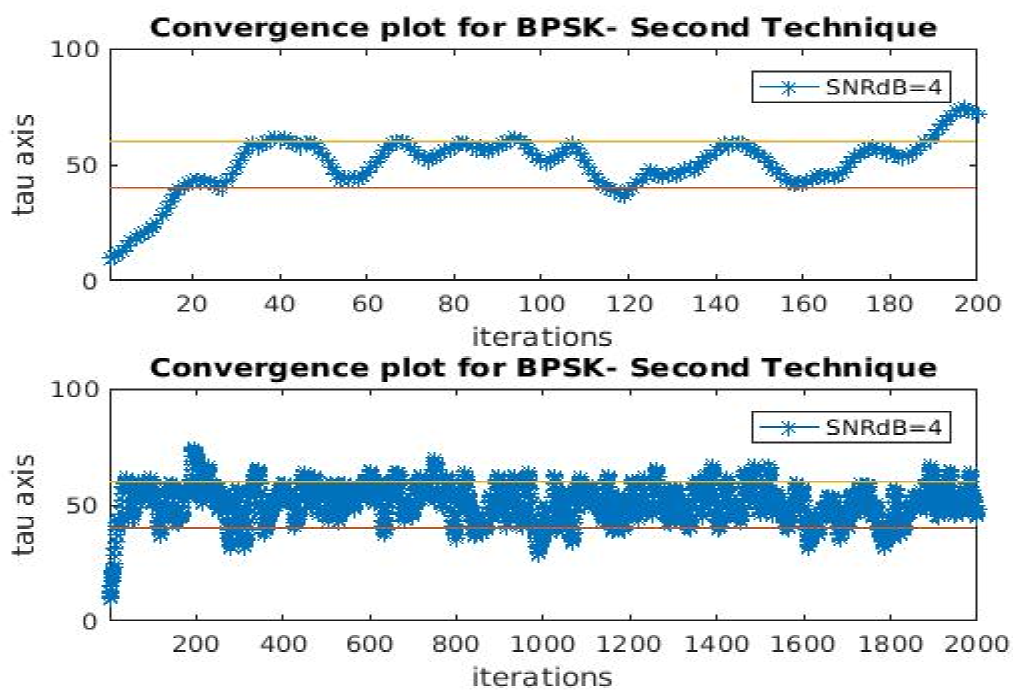

Figure 3.21: BPSK-second technique, SNR=4 dB

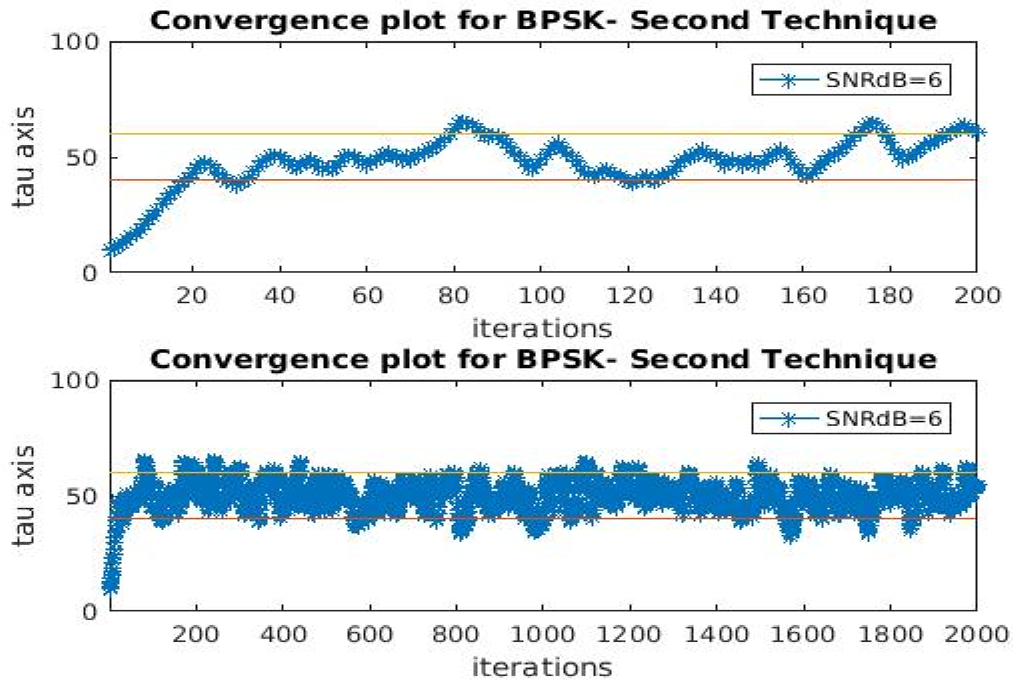

Figure 3.22: BPSK-second technique, SNR=6 dB

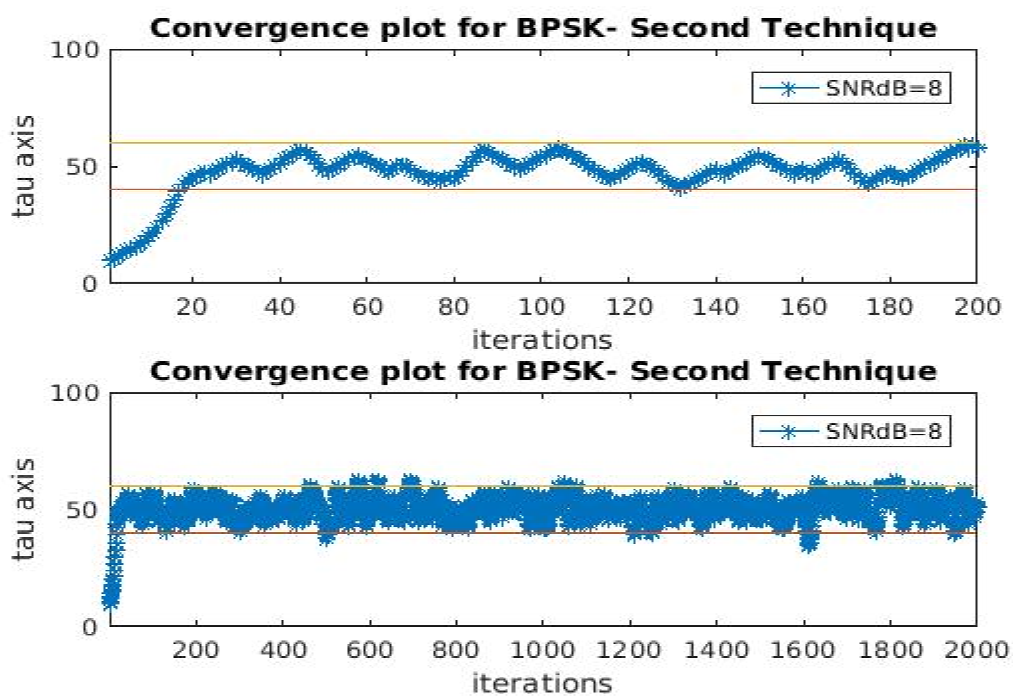

Figure 3.23: BPSK-second technique, SNR=8 dB

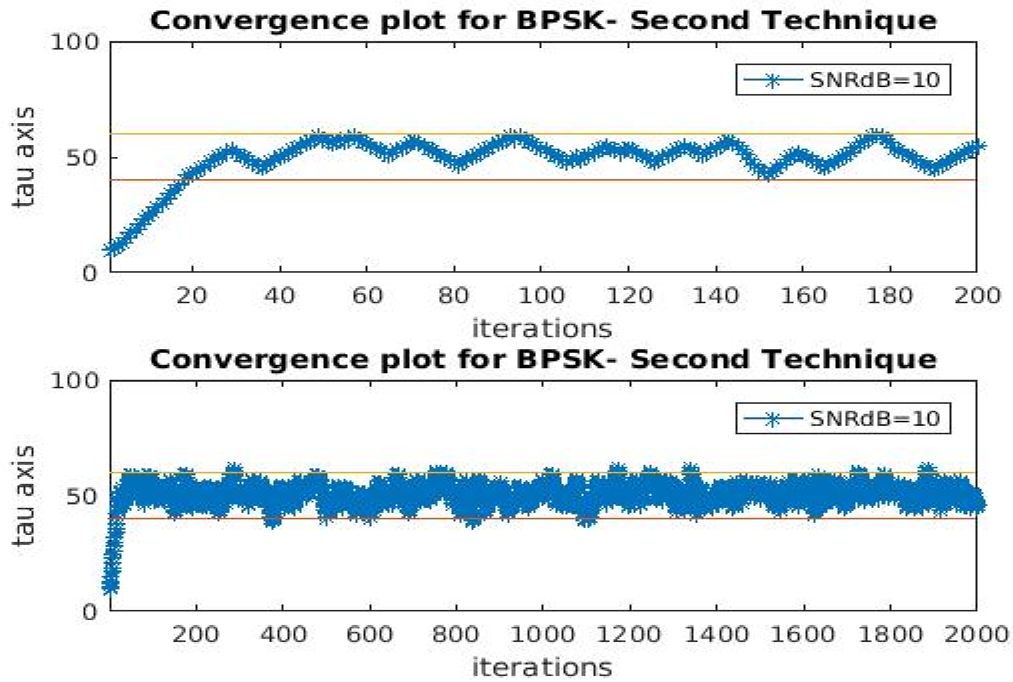

Figure 3.24: BPSK-second technique, SNR=10 dB

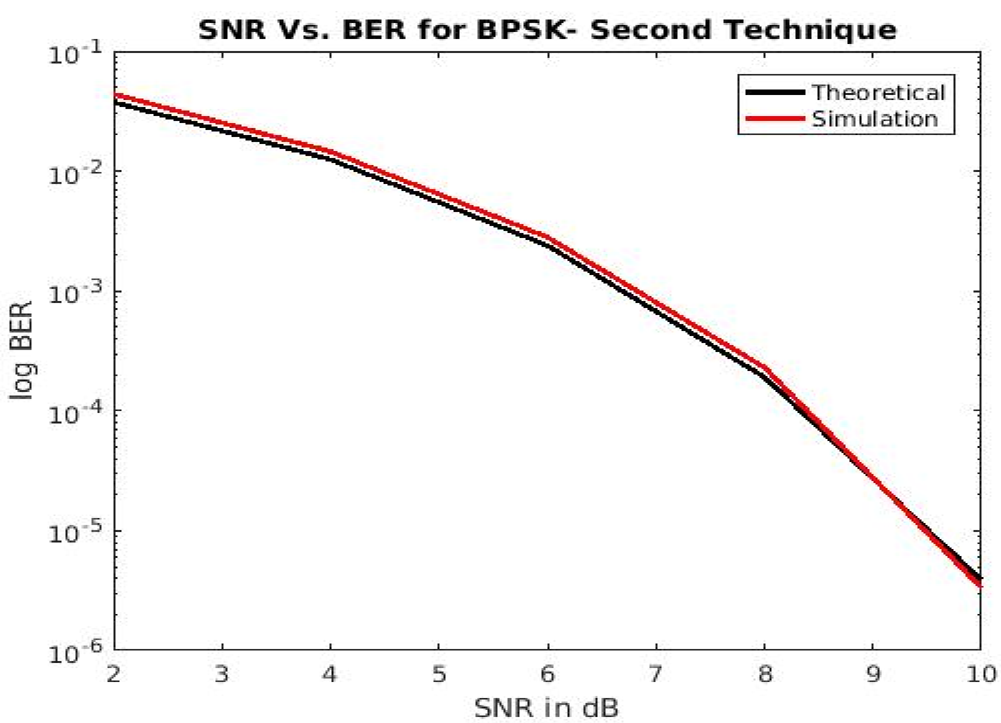

Figure 3.25: BPSK-second technique, SNR vs. BER plot

The next table reveals the MSE values that are corresponding to the SNR values for the second technique.

SNR	2 dB	4 dB	6 dB	8 dB	10 dB
MSE	71.8016	56.4763	32.2742	23.6247	19.0516

Table 3.3: states MSE values with SNR values- second technique

3.9.1 Evaluation of the second technique

This section discusses the comparison between the results of the second technique and the original results of the Gardner technique. The comparison is done by depending on four factors which are: the speed of the convergence plots, SNR vs. BER plots, the Mean Squared Error (MSE), and the complexities of the digital receiver design.

In term of the convergence speed, this technique increases the speed of the convergence. In term of the Bits Error Rate (BER), figure 3.25 states that the SNR vs. BER plot of the second technique has the same SNR vs. BER plot of the Gardner technique shown in figure 3.10. In term of MSE, the values of MSE of the second technique are worse than the MSE values of the original Gardner technique. Although the MSE values of the second technique are worse, the BER results are the same as the BER results of the Gardner technique. In terms of the design complexity, the second technique uses an additional filter which is the weighted filter. Practically, any additional component may increase the processing time in the communication system. The impact of the noise can be noticed as an oscillation in convergence plots.

3.10 The third technique to improve the timing recovery (part #1)

This code that is used in this technique is the same code that is used in the Gardner technique. The difference is the average filter, which is used by the Gardner technique, is not used in the third technique. In other words, the result of Gardener's algorithm is directly used in the loop filter without taking the mean of several instantaneous timing estimates. Also, the step size that is used in the third technique (part #1) is equal to 2. The only reason for doing this is to increase the convergence speed. It is true that this technique may not be able to sample at the optimal value all the time, but the third technique (part #1) can sample close to the optimal value. When samples are taken at or close to the optimal value, they are less affected by the noise. So, even if samples are taken close to the optimal value, these samples still have the ability to represent the actual data. The following code shows that:

third_technique.m: Error detection and loop filter.

```
%% Error detection
        sub=latesample-earlysample;   %Subtraction process
        GA=sub*midsample;
                                      %Gardner Algorithm
%% Loop filter
if mean(GA) > 0
   tau = -stepsize;                   %Shift by decreasing
else
   tau = stepsize;                    %Shift by increasing
end
```

The following plots demonstrate the convergence plots and the SNR vs. BER plot:

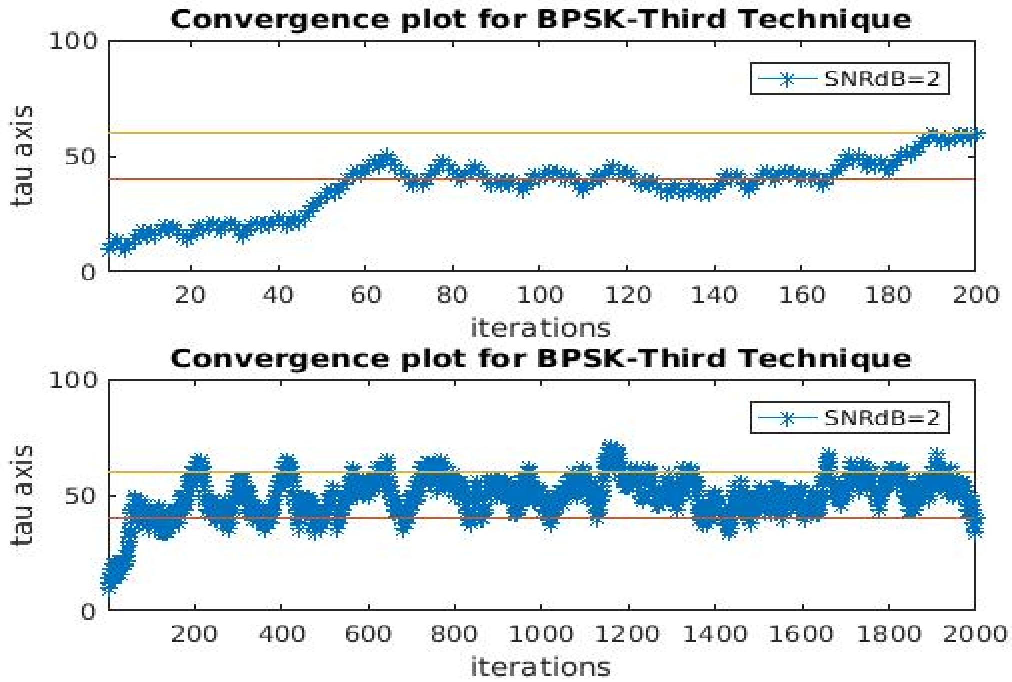

Figure 3.26: BPSK- third technique (part #1), SNR=2 dB

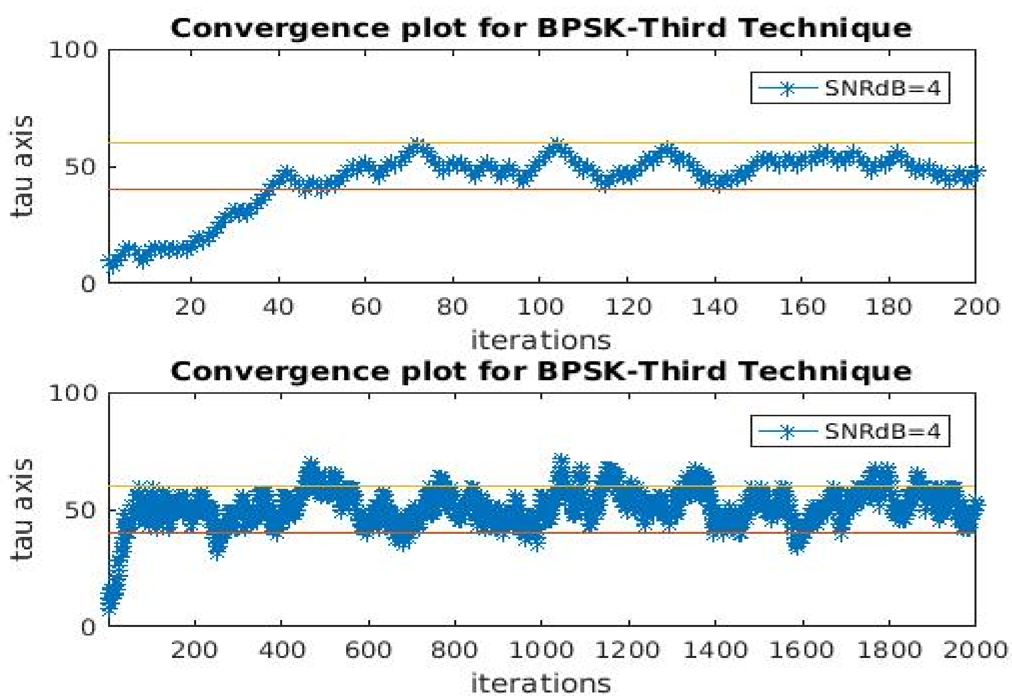

Figure 3.27: BPSK- third technique (part #1), SNR=4 dB

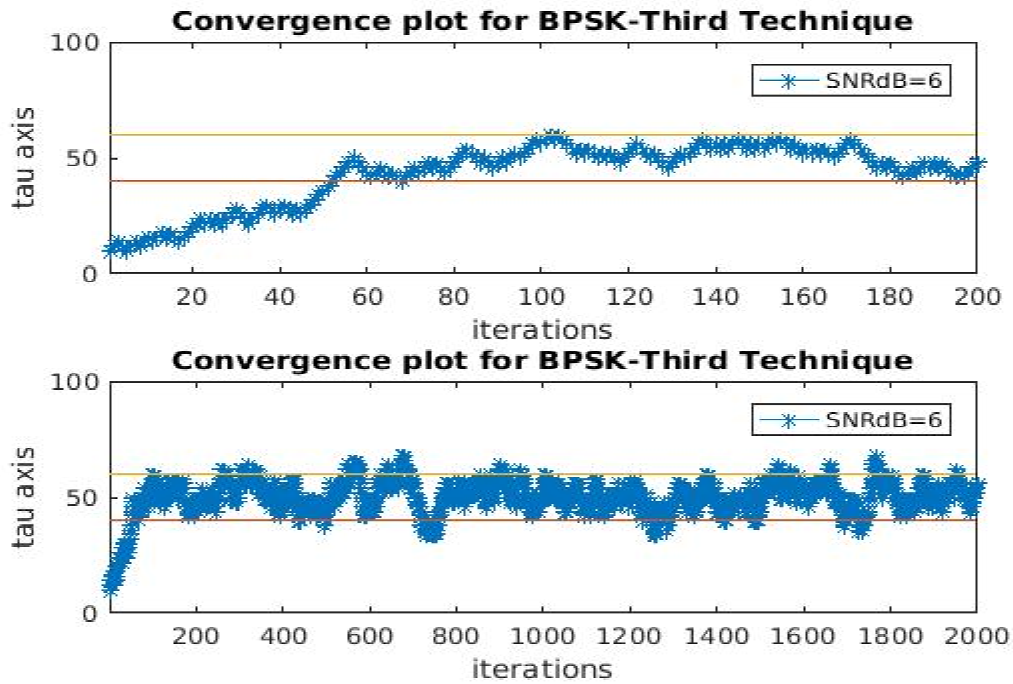

Figure 3.28: BPSK- third technique (part #1), SNR=6 dB

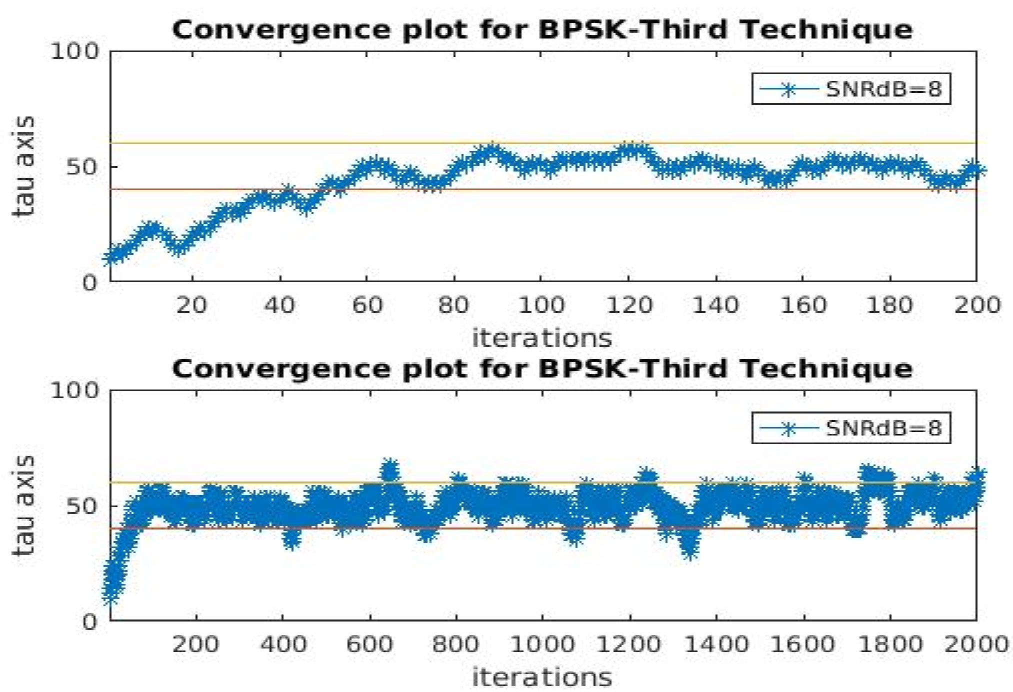

Figure 3.29: BPSK- third technique (part #1), SNR=8 dB

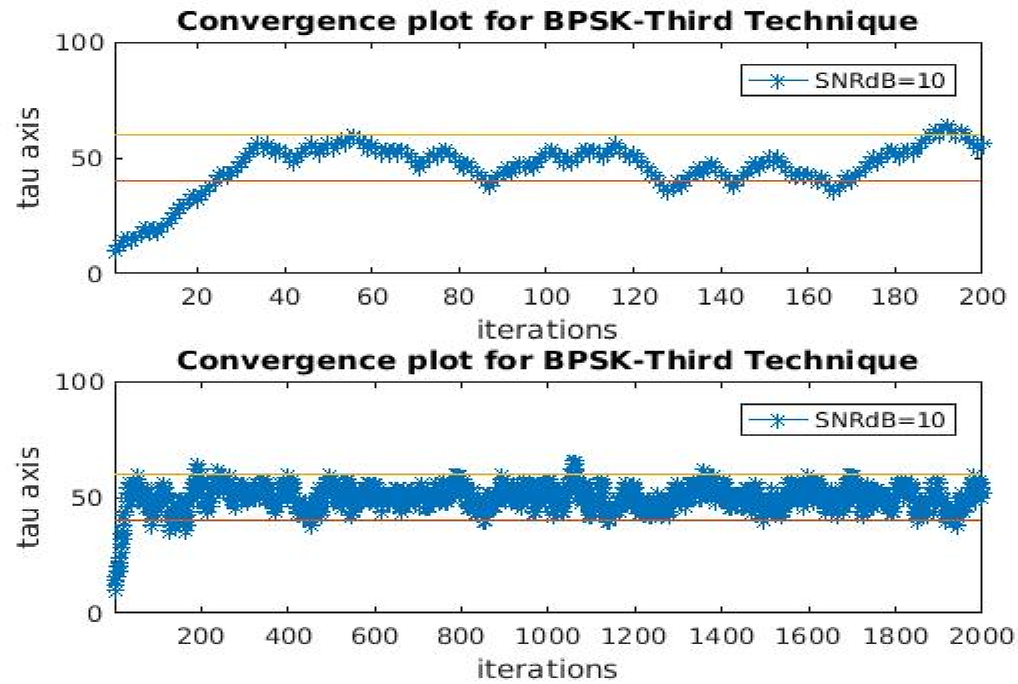

Figure 3.30: BPSK- third technique (part #1), SNR=10 dB

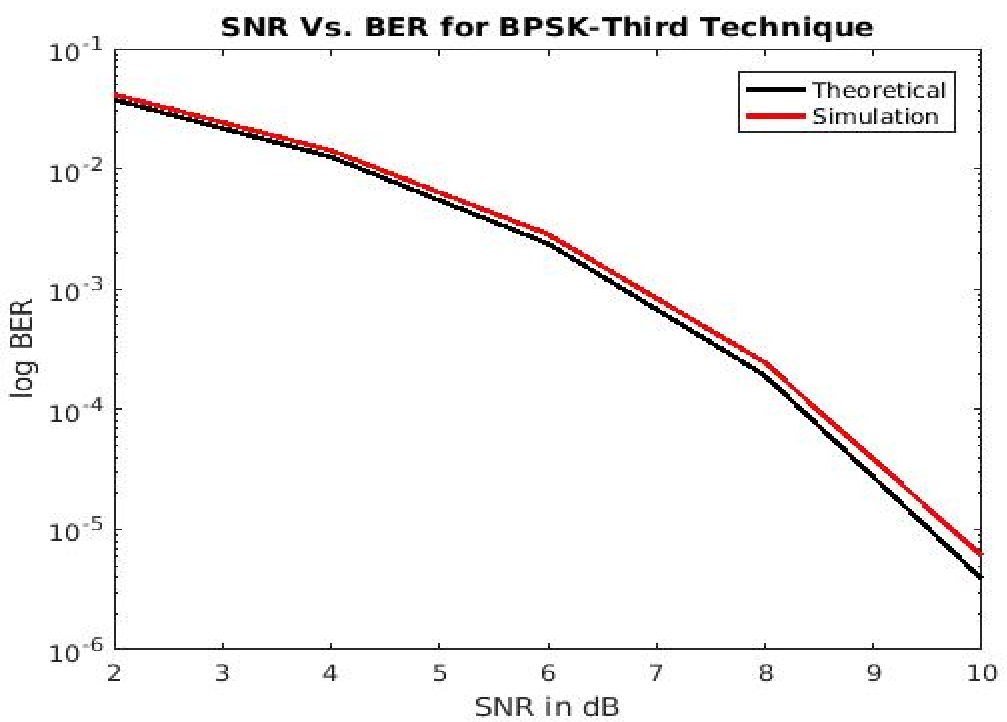

Figure 3.31: BPSK- third technique (part #1), SNR vs. BER plot

The next table reveals the MSE values that are corresponding to the SNR values for the third technique.

SNR	2 dB	4 dB	6 dB	8 dB	10 dB
MSE	46.9853	44.1305	33.9242	21.5200	20.0042

Table 3.4: states MSE values with SNR values- third technique (part #1)

3.10.1 Evaluation of the third technique (part #1)

The results of the third technique are compared with results of the Gardner technique to evaluate the performance of the third technique. The convergence takes shorter to happen than the convergence of the baseline technique. The convergence happens faster in the third technique because step size is equal to 2. Generally, the SNR vs. BER plot of the third technique has the same performance of SNR vs. BER plot of the Gardner technique (see figures 3.10 and 3.31). In term of MSE, the values of the MSE of the third technique are less than the values of the MSE of the Gardner technique (see table 3.4). In term of the complexity, the third technique does not require the average filter which reduces the processing time. As a result, reducing additional filters reduces the complexity and improves the performance of the digital receiver.

3.11 The third technique to improve the timing recovery (part #2)

The code that is used in technique is exactly the same technique that is used in the third technique (part #1). The only difference in this technique is the step size is equal to 1. This difference in the current technique influences its results. The following figures states the convergence plots and the SNR vs. BER plot of the third technique (part #2):

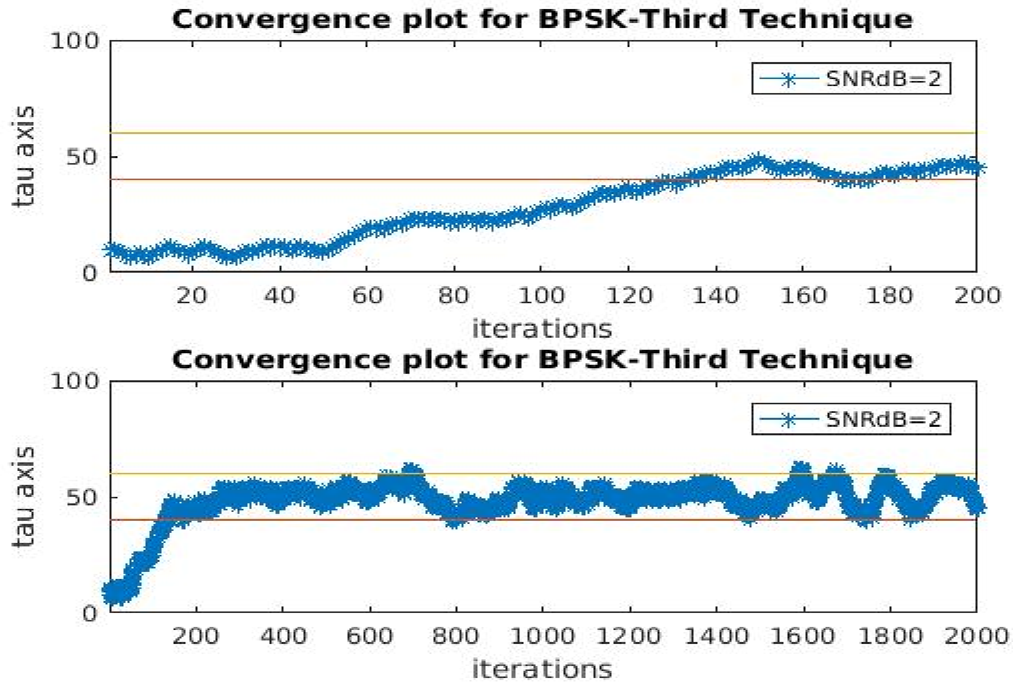

Figure 3.32: BPSK- third technique (part #2), SNR=2 dB

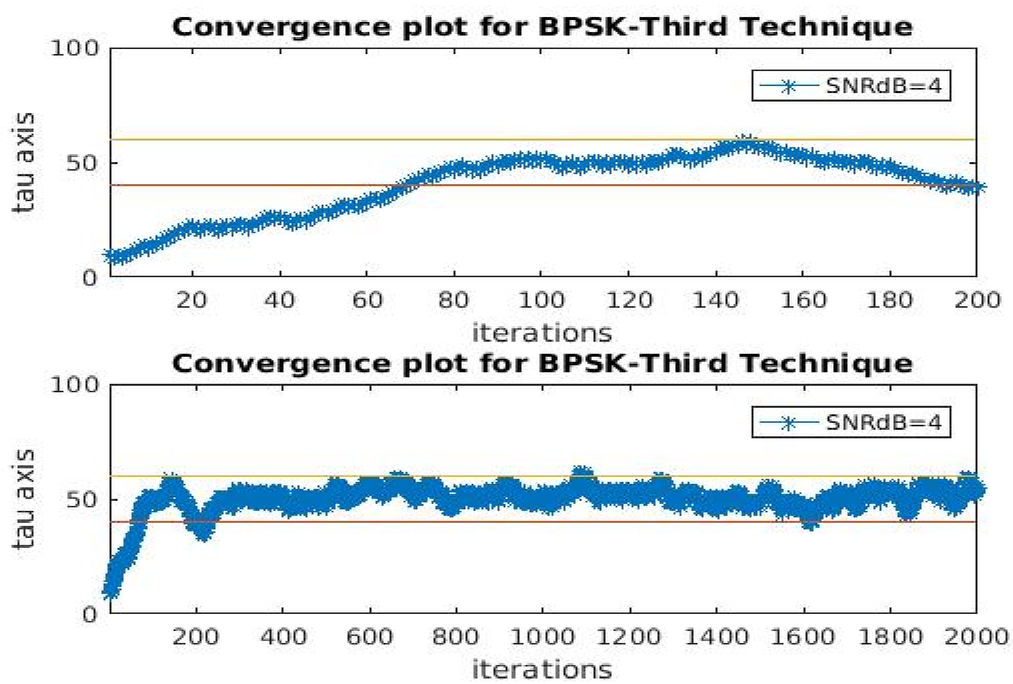

Figure 3.33: BPSK- third technique (part #2), SNR=4 dB

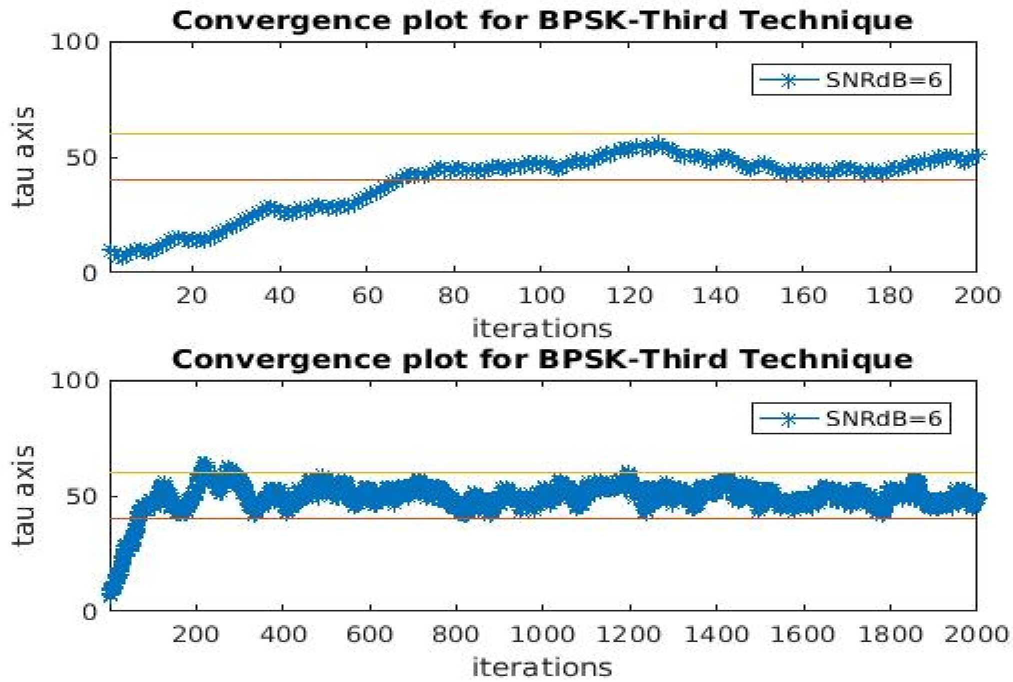

Figure 3.34: BPSK- third technique (part #2), SNR=6 dB

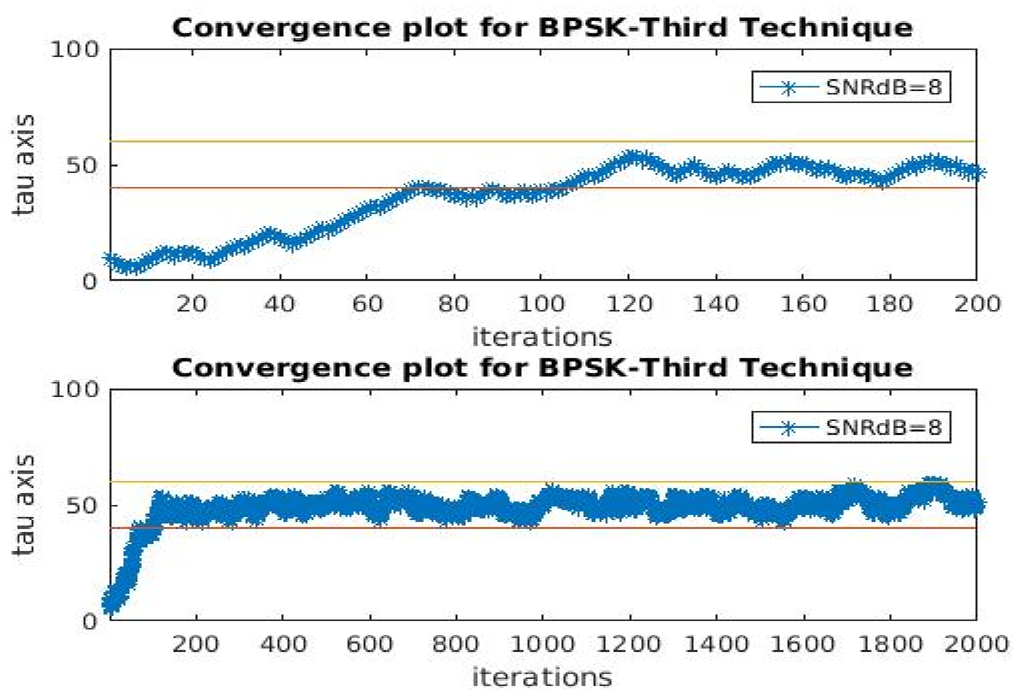

Figure 3.35: BPSK- third technique (part #2), SNR=8 dB

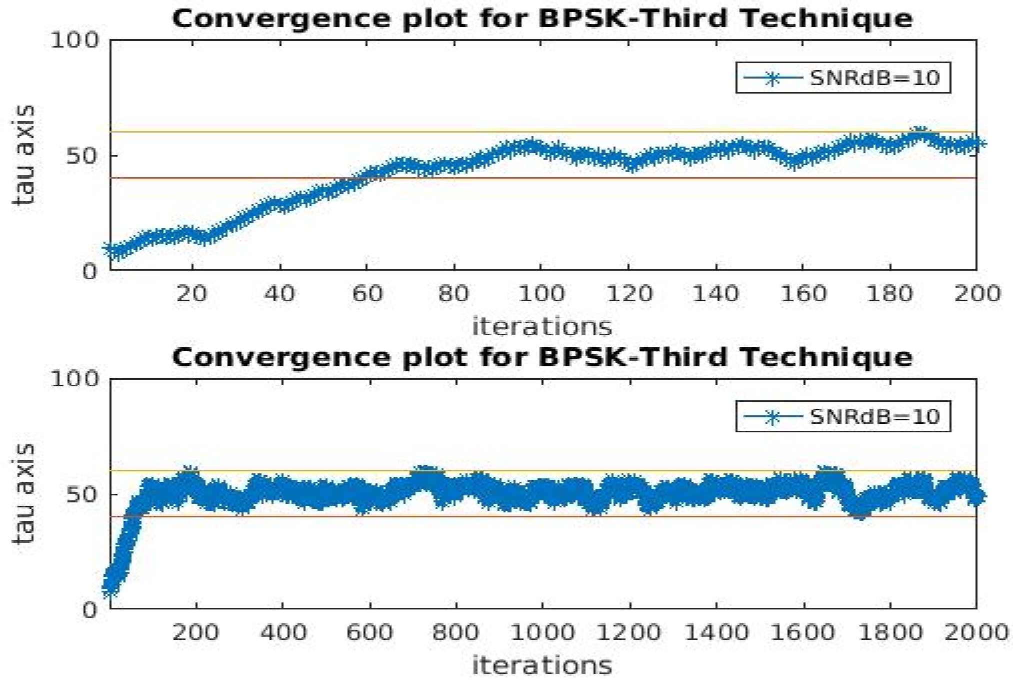

Figure 3.36: BPSK- third technique (part #2), SNR=10 dB

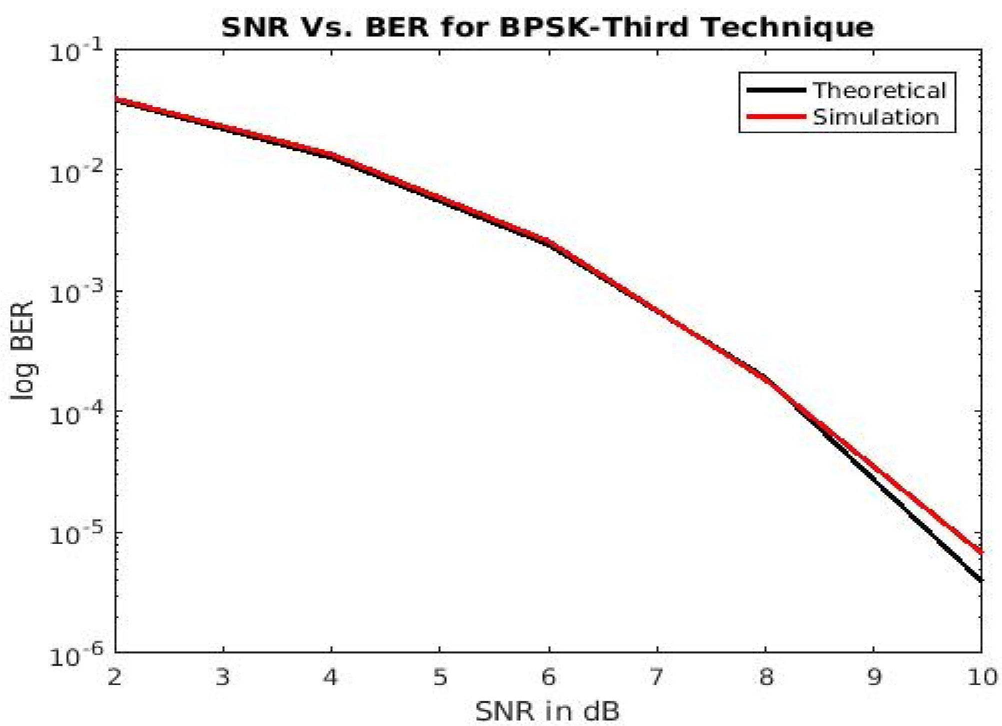

Figure 3.37: BPSK- third technique (part #2), SNR vs. BER plot

The next table reveals the MSE values that are corresponding to the SNR values for the third technique.

SNR	2 dB	4 dB	6 dB	8 dB	10 dB
MSE	23.5547	20.3295	15.5358	13.1800	12.3379

Table 3.5: states MSE values with SNR values- third technique (part #2)

3.11.1 Evaluation of the third technique (part #2)

The results of the third technique (part #2) are compared with results of the Gardner technique to evaluate the performance of the third technique. The convergence takes longer than the convergence of the baseline technique. The convergence happens late in the third technique (part #1) because the step size is equal to 1. Generally, the SNR vs. BER plot of this technique has the same performance of SNR vs. BER plot of the Gardner technique (see figures 3.10 and 3.37). In term of MSE, the values of the MSE of the third technique (part #2) are less than the values of the MSE of the Gardner technique (see table 3.5). In term of the complexity, this technique does not require the average filter which reduces the processing time. As a result, reducing additional filters reduces the complexity and improves the performance of the digital receiver.

It is important to demonstrate that each part of the third technique has different features in terms of the convergence plots and MSE values. The first part of the third technique has a faster convergence than those of the Gardner technique, but it has worse MSE values. The second part of the third technique has slow convergence compared to the Gardner technique. However, it has low MSE values. As a result, it kind of trade off, and it depends on the requirements and circumstances of wireless communication systems.

For example, when the sinc wave is used in the pulse shape code, this wave is more affected by MSE values which may negatively reflect on BER performance. So, it is better to choose the second part of the third technique. However, the pulse shape that is used in the codes of this thesis is the half sine wave which is less affected by MSE values. In other words, the range of MSE values for the third technique does not have disadvantage on the BER performance of this technique. Therefore in this thesis, the first part of the third technique which has fast convergence is used further with QPSK - third technique and with QPSK - Alamouti - Gardner (QAG) third technique.

3.12 The fourth technique to improve the timing recovery

The fourth technique uses the same code that is used by the Gardner technique. In this technique, the average filter is not used in the decision operation. So basically, it is looks like the third technique, but this technique uses another filter which is the weighted filter. In other words, there are 11 values of tau (-1,-3,-5,-7,-9,1,3,5,7,9,& 0) that depends on the value of the Gardner algorithm. This way of using different step sizes makes the convergence happens faster. The following code shows that:

fourth_technique.m: Error detection and loop filter.

```
%% Error detection
 sub=latesample-earlysample;                    %Subtraction process
 GA=sub*midsample;                              %Gardner Algorithm
 %% Loop filter
 if GA>0 && GA<0.5
     tau=-1;
 elseif GA>0.5 && GA<1
     tau=-3;
 elseif GA>1 && GA<1.5
     tau=-5;
 elseif GA>1.5 && GA<2
     tau=-7;
 elseif GA>2
     tau=-9;
 elseif GA<0 && GA>-0.5
     tau=1;
 elseif GA<-0.5 && GA>-1
     tau=3;
 elseif GA<-1 && GA>-1.5
```

```
    tau=5;
elseif GA<-1.5 && GA>-2
    tau=7;
elseif GA<-2
    tau=9;
else
    tau=0;
end
```

The following plots are the convergence plots and SNR vs. BER plot of the fourth technique:

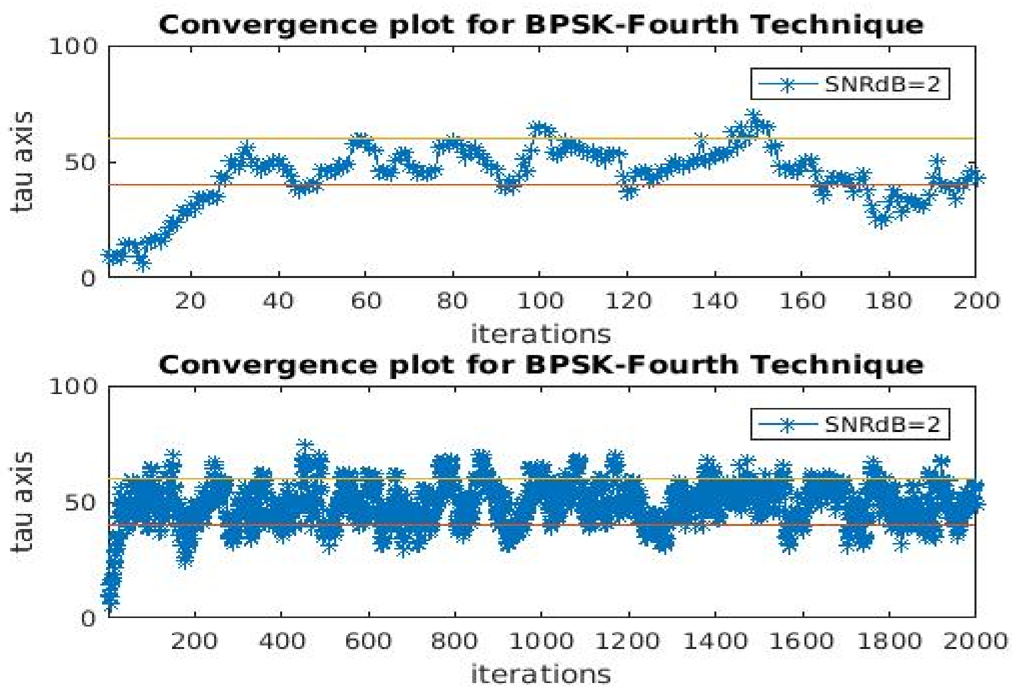

Figure 3.38: BPSK- fourth technique, SNR=2 dB

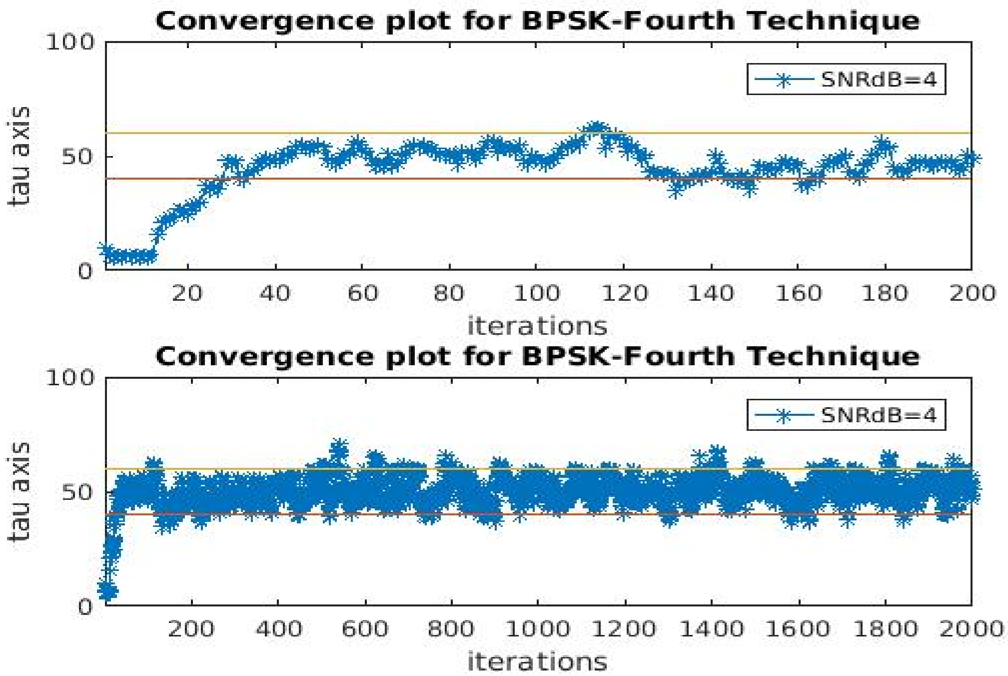

Figure 3.39: BPSK- fourth technique, SNR=4 dB

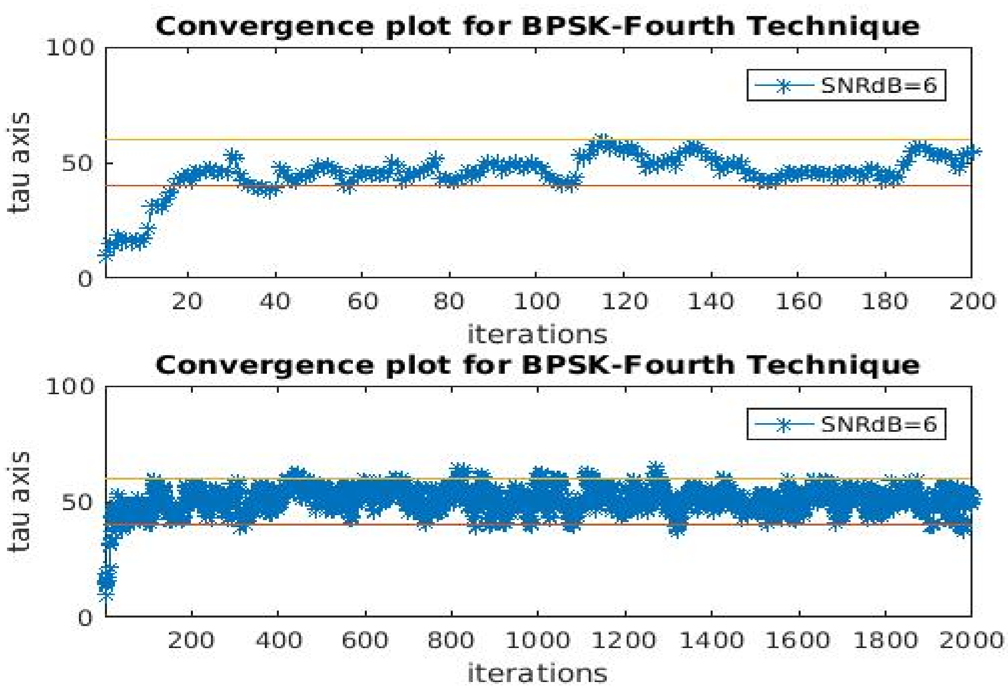

Figure 3.40: BPSK- fourth technique, SNR=6 dB

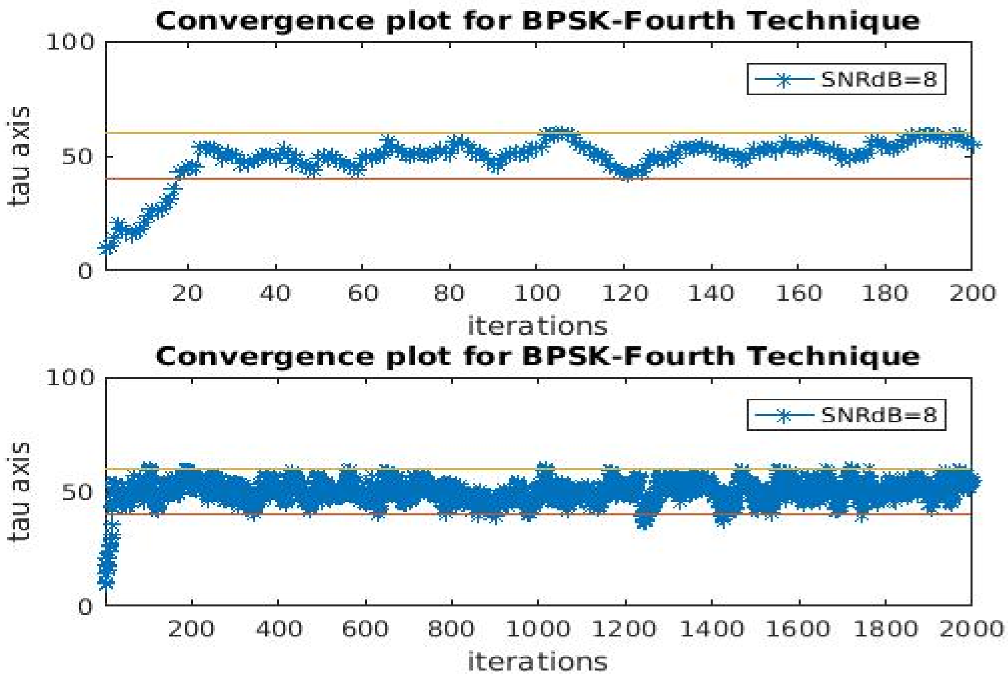

Figure 3.41: BPSK- fourth technique, SNR=8 dB

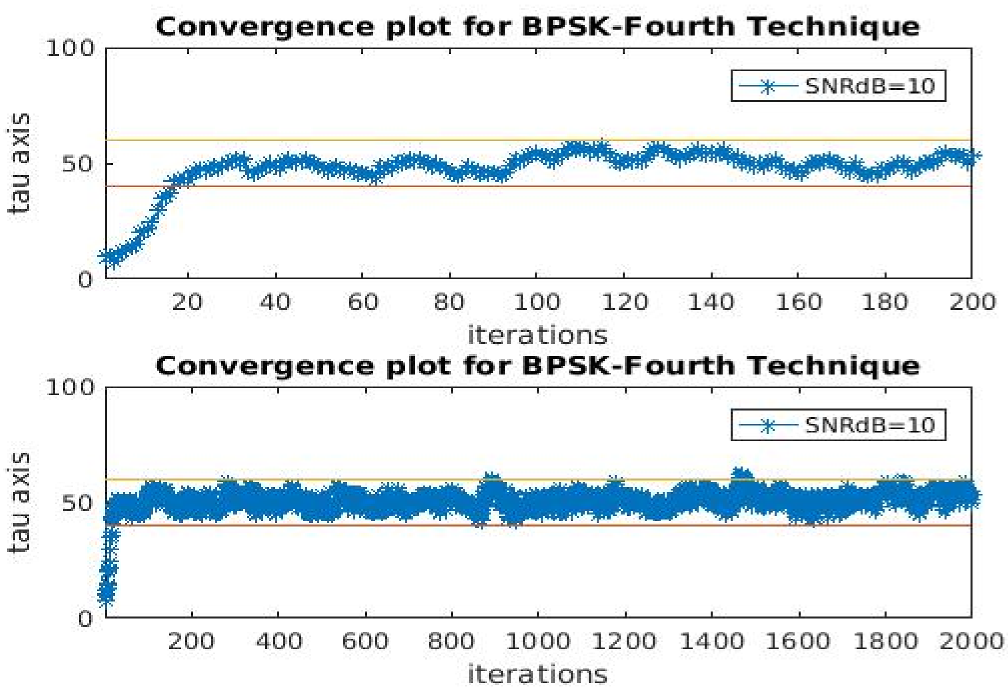

Figure 3.42: BPSK- fourth technique, SNR=10 dB

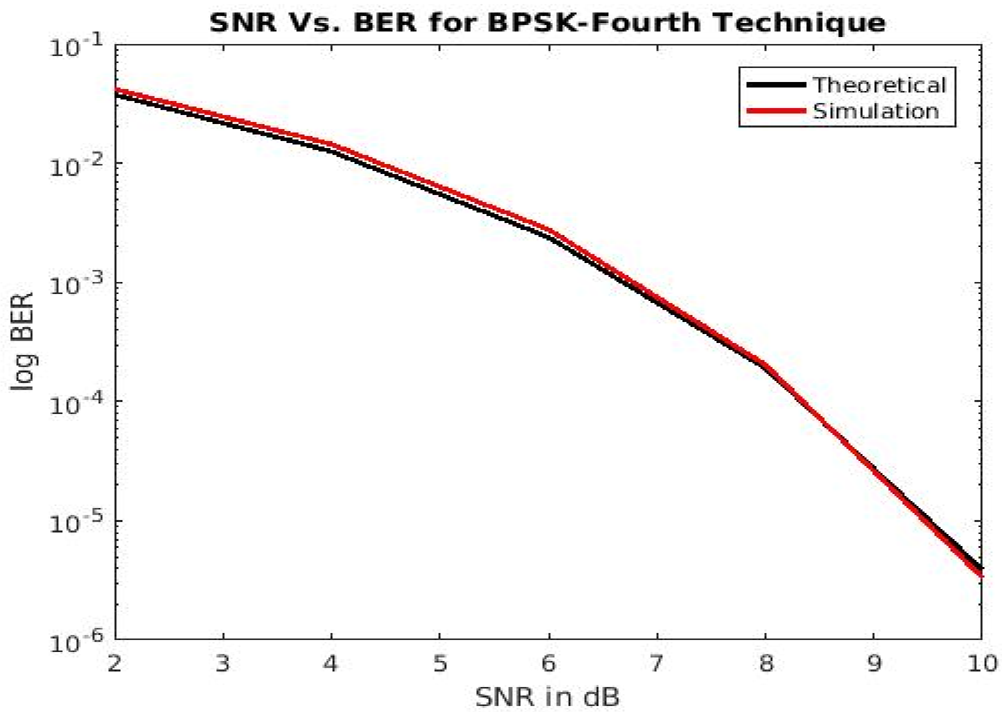

Figure 3.43: BPSK- fourth technique, SNR vs. BER plot

The next table reveals the MSE values that are corresponding to the SNR values for the fourth technique.

SNR	2 dB	4 dB	6 dB	8 dB	10 dB
MSE	67.7147	33.7611	23.8284	15.1547	11.3926

Table 3.6: states MSE values with SNR values- fourth technique

3.12.1 Evaluation of the fourth technique

In the light of the results, the convergence plots of the fourth technique have a faster convergence, and this is consistent with the work of the weighted filter. In other words, using different value for the tau increases the speed of the convergence. Also, the SNR vs. BER of the fourth technique has the same behavior of the SNR vs. BER of the Gardner technique (see figure 3.10 and 3.43). In term of the MSE, the fourth technique has higher MSE values than the Gardner technique, especially when the SNR is low or when the noise is high.

Although the MSE values of the fourth technique are high, the BER results still the same as the BER results of the Gardner technique. In term of the complexity, the fourth technique removes the average filter and uses the weighted filter. It is supposed that the complexity which results from using the weighted filter is less than the complexity of using average filter. However, the practical implementation can show which technique is less complexity.

3.13 The fifth technique to improve the timing recovery

The fifth technique uses the code of the Gardner technique with some changes in using the average filter. This technique removes the normal average filter that is used to find the mean of several instantaneous timing estimates. However, this technique uses a new average filter to find the mean for midway samples. Also, the fifth technique involves the weighted filter to indicate the value of tau. This weighted filter provides nine values which are (1,3,5,7,-1,-3,-5,-7,& 0) that can be used depending on the value of the Gardner algorithm. The following Matlab code illustrates that:

---

fifth_technique.m: Error detection and loop filter.

```
    midsample=mean(received(center-1:center+1));%The midway sample
    latesample=received(center+delta);          %The late sample
    earlysample=received(center-delta);         %The early sample
    a(rit)=earlysample;                         %Save samples
    %% Error detection
    sub=latesample-earlysample;                 %Subtraction process
    GA=sub*midsample;                           %Gardner Algorithm
    %% Loop filter
    if GA<=0.2 && GA>=0
        tau=-1;
    elseif GA<=0.5 && GA>0.2
        tau=-3;
    elseif GA<=1 && GA>0.5
        tau=-5;
    elseif GA>1
        tau=-7;
```

```
elseif GA>=-0.2 && GA<0
    tau=1;
elseif GA>=-0.5 && GA<-0.2
    tau=3;
elseif GA>=-1 && GA<-0.5
    tau=5;
elseif GA <-1
    tau=7;
else
    tau=0;
end
```

The following plots show the convergence plots and SNR vs. BER plot of the fifth technique:

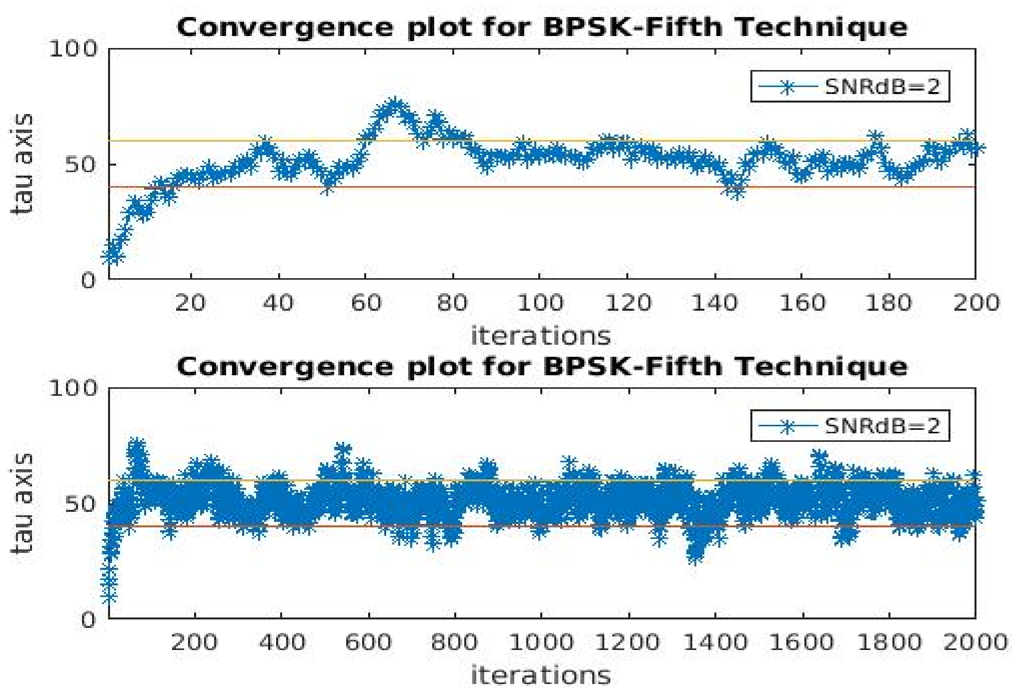

Figure 3.44: BPSK- fifth technique, SNR=2 dB

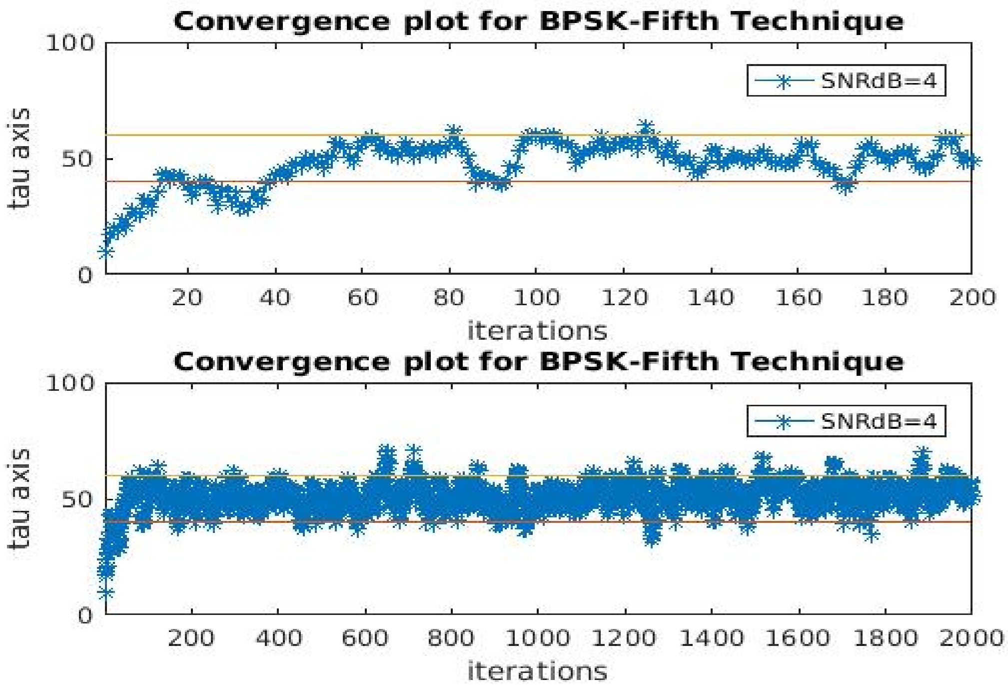

Figure 3.45: BPSK- fifth technique, SNR=4 dB

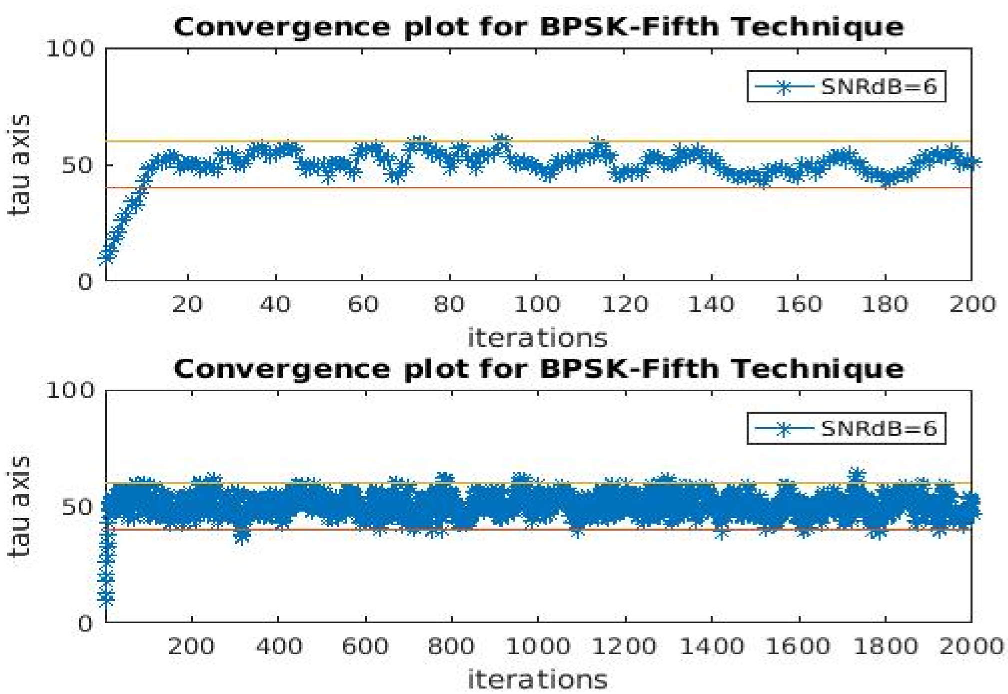

Figure 3.46: BPSK- fifth technique, SNR=6 dB

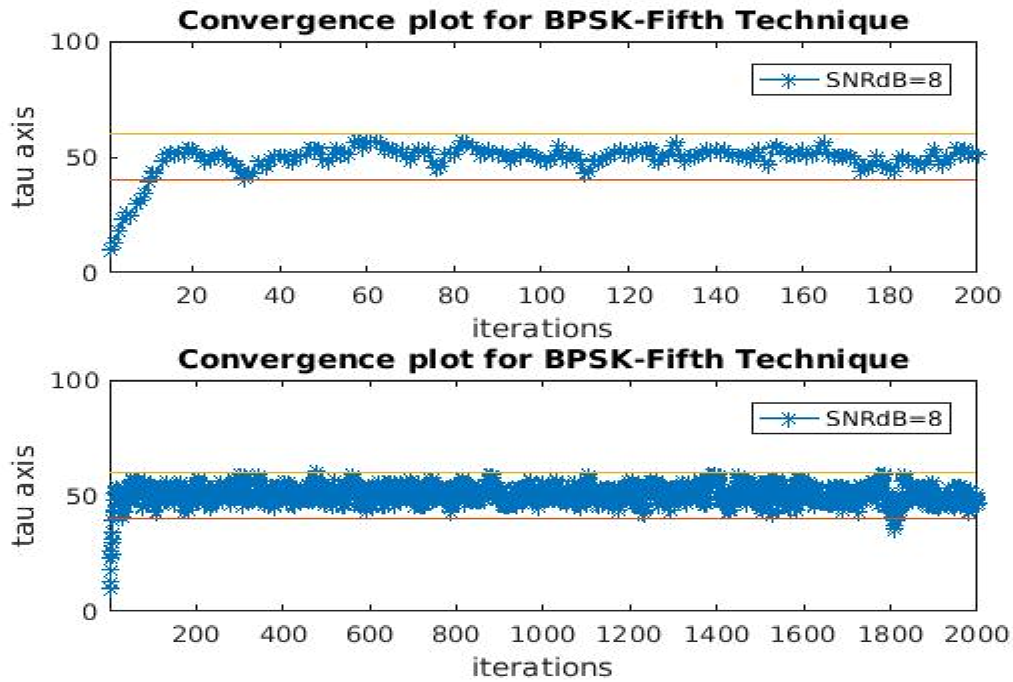

Figure 3.47: BPSK- fifth technique, SNR=8 dB

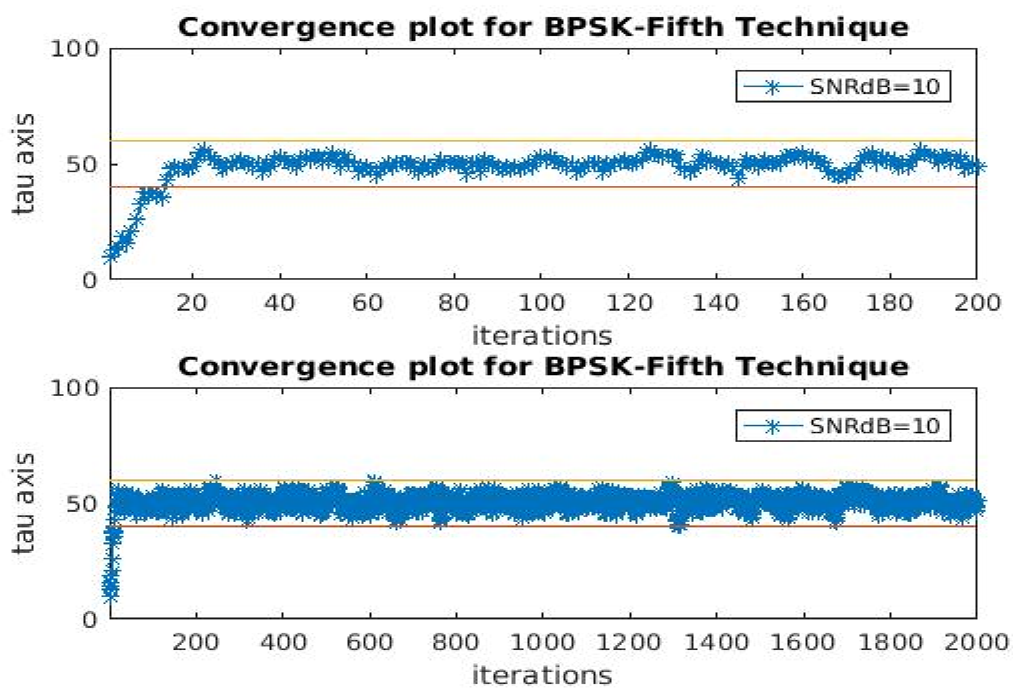

Figure 3.48: BPSK- fifth technique, SNR=10 dB

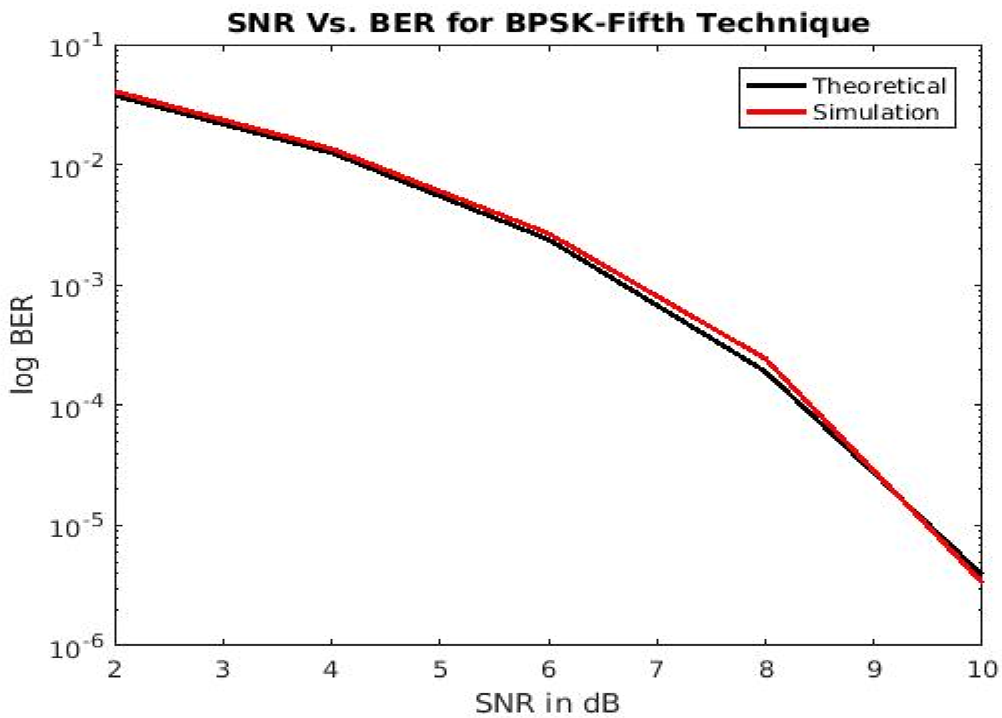

Figure 3.49: BPSK- fifth technique, SNR vs. BER plot

The next table reveals the MSE values that are corresponding to the SNR values for the fifth technique.

SNR	2 dB	4 dB	6 dB	8 dB	10 dB
MSE	33.6558	21.4411	17.4874	14.127	13.6579

Table 3.7: states MSE values with SNR values- fifth technique

3.13.1 Evaluation of the fifth technique

This technique can be evaluated by comparing its results with the Gardner technique results. In term of convergence plots, the fifth technique has a faster convergence than the Gardner technique. The convergence happens faster because the weighted filter is used which gives different values of tau with larger step sizes. Additionally, the SNR vs. BER plot of the fifth technique seems to be the same as the SNR vs. BER plot of the Gardner technique. In term of the

MSE, the fifth technique has less MSE values than the Gardner technique. In the light of the complexity, the fifth technique eliminates the normal average filter that is used in the Gardner technique. On the other hand, this technique uses a new average filter to find the mean for midway samples. In addition to that, a weighted filter is used in this technique. As a result, it is expected that the complexity of the digital receiver structure increases in the fifth technique.

3.14 Summary and comparison of the five techniques-BPSK

It is worthwhile to summarize the five techniques - BPSK that are introduced in the Thesis. Moreover, the summary gives a clear picture about the benefits and features each technique. In order to facilitate the process of comparison, the convergence plots when SNR is equal to (10 dB) are be included. Furthermore, the tables of Mean Squared Error (MSE) are stated for each technique.

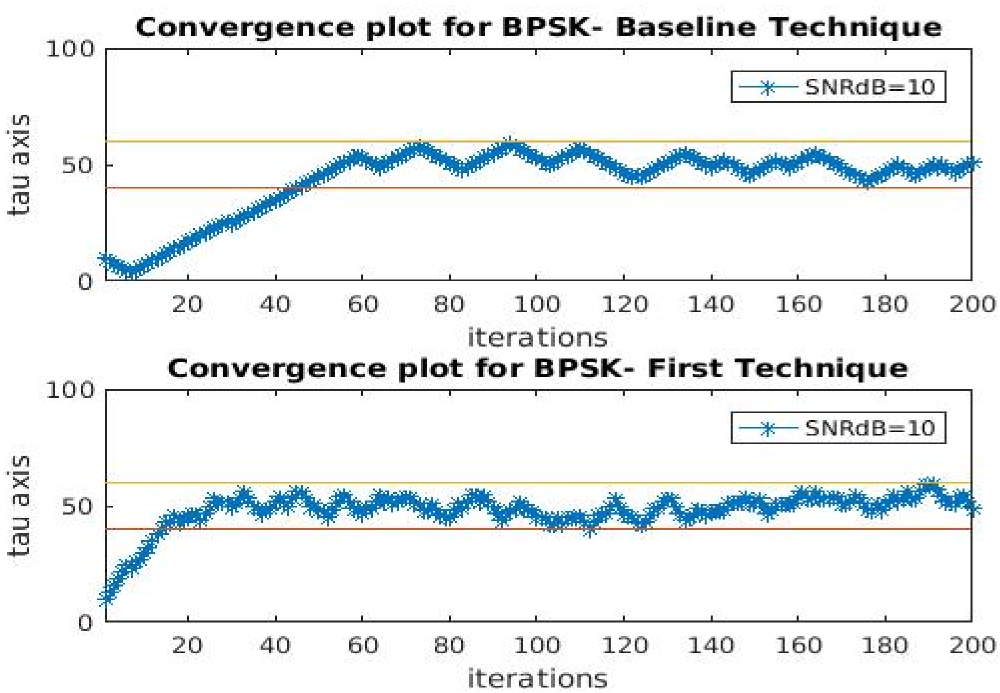

Figure 3.50: BPSK- Baseline technique and first technique, SNR=10 dB

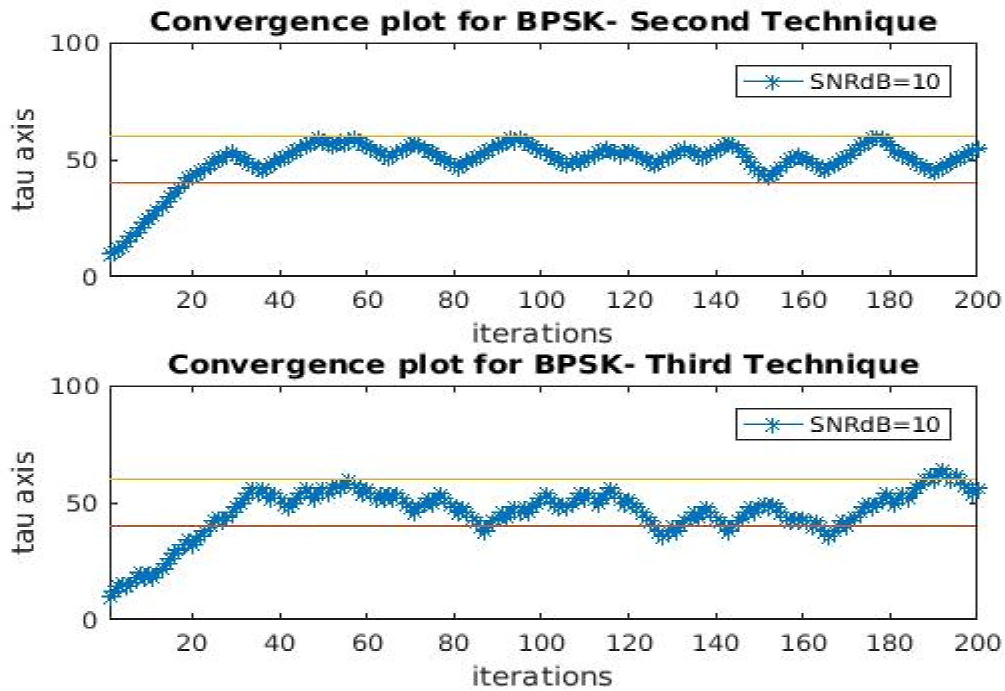

Figure 3.51: BPSK- second technique and third technique, SNR=10 dB

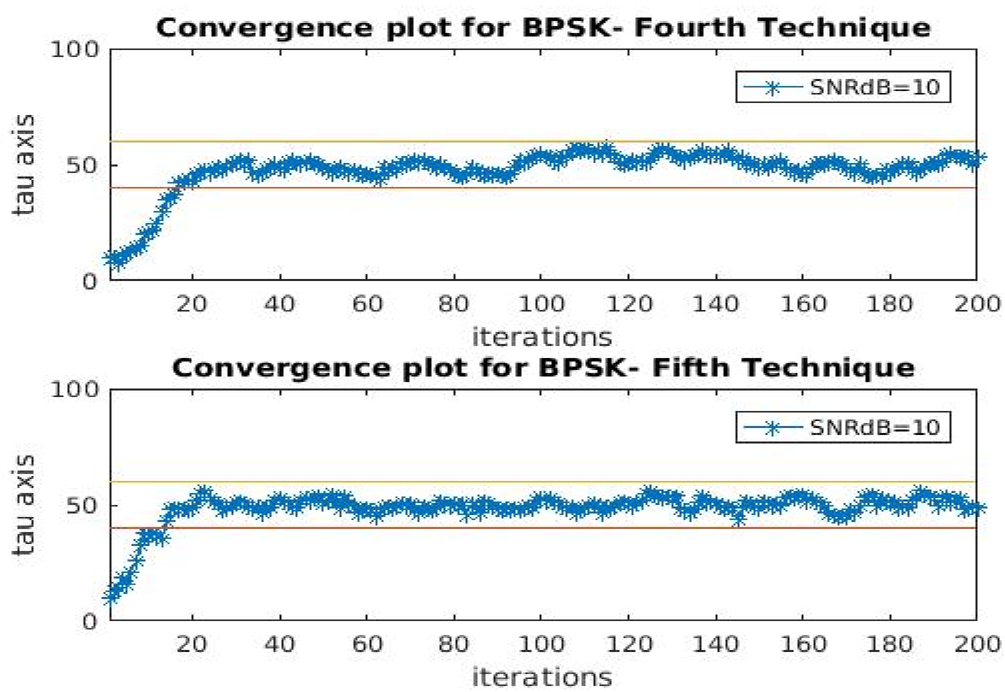

Figure 3.52: BPSK- fourth technique and fifth technique, SNR=10 dB

SNR	2 dB	4 dB	6 dB	8 dB	10 dB
MSE_Gardner technique	36.9274	28.8579	23.5400	18.2811	13.9632
MSE_first technique	54.3821	49.5479	31.5684	24.3621	16.7868
MSE_second technique	71.8016	56.4763	32.2742	23.6247	19.0516
MSE_third technique	46.9853	44.1305	33.9242	21.5200	20.0042
MSE_fourth technique	67.7147	33.7611	23.8284	15.1547	11.3926
MSE_fifth technique	33.6558	21.4411	17.4874	14.127	13.6579

Table 3.8: Summary of MSE of all techniques for BPSK-Gardner technique

3.15 Baseline Matlab code of Gardner technique with QPSK

The QPSK modulation scheme is used to transmit the data amount which is (15×10^5). This amount of data represents the number of samples. It is given that each sample has two bits, so the total number of bits is (3×10^6) which is the same number of bits that are used in BPSK - Gardner technique. The following code shows the data generation and pulse shape:

Baseline_GardnerQPSK.m: Data generation and the pulse shape.

```
%% Data generation
N=15*10^5;                          % Amount of data
Qpsk=[1+1i 1-1i -1+1i -1-1i];       % Four possible complex No. for QPSK
data= Qpsk(randi(4,1,N));           % The data
Tsym=100;                           % No.of samples per symbol
noise=(randn(1,length(data)*Tsym)+1i*randn(1,length(data)*Tsym));
                                    %Generating random numbers for n1
%% pulse shape
p = sin(2*pi*(0:Tsym-1)/(2*Tsym));%Sinusoidal wave
```

```
data_up = zeros(1,length(data)*Tsym);%Creation a memory of zeros
data_up(1:Tsym:end) = data;         %Interpolation the data
S11 = conv(data_up,p);              %The convolution operation
S1=S11(1:end-99);                   %Remove last 99 bits that are
                                    %added due to the convolution
```

After the pulse shape is done, the noise is added to the transmitted signal. The noise is represented by the Signal to Noise Ratio (SNR). The formula that is used by Gardner to detect the error, involves using both the In-phase and Quadrature channels. As a result, the QPSK-Gardner technique is less affected by the noise. Consequently, the range of the SNR values, which is used in this technique, is from 2 dB to the 10 dB. Then, the SNR values are converted into linear values. After that, a loop is created, and it is repeated for each SNR value to find values of noise that are added to the transmitted signal. After adding the noise, the QPSK bits are converted from complex numbers to normal numbers to conduct the error detection process on them.

Then, the transmitted signal is received at the receiver, and the detection and correction process starts. This section starts with giving information about initial values. According to the Gardner technique, the optimal value for midway samples should be at n*100, where n is the sequence of symbol in data. The first midway sample (called `center` in the code) is assumed to be received at 60. In addition to midway sample, the Gardner algorithm involves to find the early sample and the late sample. The early and late samples can be calculated by finding the value of samples at (`center+delta`) and (`center-delta`) respectively. The value of `delta` is equal to the half symbol period which is equal to (Tsym/2=50).

After applying the Gardner algorithm, the shift value, that is used to correct the sampling operation, depends on the finding the mean of several instantaneous timing estimates. In this code, the number of the value, that is taken to find the average, is assumed to be equal to six (called avgsamples in the code). The Gardner technique supposes that the correction depends on the sign of the mean more than the value itself [1], so the step size is assumed to be equal to 1.

When the initial values are given, another inside loop is created to conduct the sampling operation, the error detection, loop filter, and the correction operations. Next, center (remind) values are saved to be used in the convergence plots. The following code demonstrates the noise addition, error detection, correction:

Baseline_GardnerQPSK.m: Noise addition, error detection and correction.

```
%% Noise addition
SNRdB=2:2:10;                % Signal to Noise Ratio
SNR=10.^(SNRdB/10);          % The linear values for the noise
for cv=1:length(SNRdB)       % Generate a loop
    sigma=sqrt(1)/sqrt(2*SNR(cv));% Sigma generation
    n1=sigma*noise;          %Noise generation
    S=S1+n1;                 % Received signal with noise

    %% Conversion complex No. to normal No.
    Sreal= real(S);          %Create a vector for real No. of S
    Simag= imag(S);          %Create a vector for imaginary No. of S

    %% Detection and correction
    tau=0;                   %Initial value for tau
    delta=Tsym/2;            %The shifting value before and after the
                             %midway sample
    center=60;               %The assumed place for the first
                             %midway sample
    a1=zeros(1,N-1);         %A memory of zeros
    a2=zeros(1,N-1);         %A memory of zeros
    cenpoint=zeros(1,N-1);   %A memory of zeros for the midway
                             %samples
    remind=zeros(1,N-1);     %A memory of zeros for the remind
    avgsamples=6;            %Six values of Gardner algorithm are
                             %used to find the average
    stepsize=1;              %Correction step size
    rit=0;                   %Iteration counter
    GA=zeros(1,avgsamples);  %A memory of zeros
    tauvector=zeros(1,1900);%A memory of zeros for tau vector
                             %(2000-100)
```

```
uor=0;                      %A counter for the tau vector
a=zeros(1,N-1);             %A memory of zeros

for ii= (Tsym/2)+1:Tsym:N*Tsym-(Tsym/2)
    rit=rit+1;                            %A counter

    %% Sampling the real part
    midsample1=Sreal(center);             %The midway sample
    latesample1=Sreal(center+delta);      %The late sample
    earlysample1=Sreal(center-delta);     %The early sample
    a1(rit)=earlysample1;                 %Save samples

    %% Sampling the imaginary part
    midsample2=Simag(center);             %The midway sample
    latesample2=Simag(center+delta);      %The late sample
    earlysample2=Simag(center-delta);     %The early sample
    a2(rit)=earlysample2;                 %Save samples

    %% Error detection
    sub1=latesample1-earlysample1;
    sub2=latesample2-earlysample2;
    GA(mod(rit,avgsamples)+1)=sub1*midsample1+sub2*midsample2;
                                          %Gardner Algorithm
    %% Loop filter
   if mean(GA) > 0
        tau = -stepsize;
    elseif mean(GA)< 0
        tau = stepsize;
   else
        tau=0;
   end

    %% Safe remind values
    cenpoint(rit)=center;              %Save positions of midway
                                       %samples
    remind(rit)=rem((center-Tsym/2),Tsym);
                                       %Save remind values to find
                                       %convergence plots
    %% tau vector
   if rit>=100 && rit<2000             %tau vector from 100 to 2000
        uor=uor+1;                     %where the convergence happens
        tauvector(uor)= (remind(rit)- (Tsym/2)).^2;%Difference
   end                                 %between remind and Tsym

    %% Correction
```

```
        center=center+Tsym+tau;          %Adding the tau value
      if center>=N*Tsym-(Tsym/2)-1       %Break the loop when the
           break;                        %midway sample reach to 51
      end                                %samples before the last
  end                                    %sample
```

Then, the tau vector is created to be used later to find the values of Mean Squared Error (MSE), and the values of samples are combined to reform the data. After the combining, the convergence behavior is plotted for each SNR value. When the convergence behavior is plotted, the BER value is calculated for each SNR. First, the total error is computed as it is stated in the following code. When the data is equal to 1 and the received sample is less than zero, this considers as an error. Similarly, when the data is equal to -1 and the received sample is more than zero, this also consider as an error. Second, the BER is calculated by dividing the total computed error over the all data amount. The loop of the SNR values ends after computing the BER. Finally, the theoretical BER is calculated, and the SNR vs. BER figure is plotted as it is showed in the following code:

Baseline_GardnerQPSK.m: MSE, convergence plot, and BER plot.

```
    %% Mean Squared Error (MSE)
    MSE(cv)=mean(tauvector);          %Finding the Mean Squared Error

    %% Combining the all bits
    for df=1:(N-1)
        a(df)=[a1(df)+a2(df)*1i];     %Combine the bits to create
    end                               %complex Numbers

    %% convergence plot
    figure
    symbols = 200;
    subplot(2,1,1);
    plot(remind(1:symbols),'*-');
    hold on
```

```
    lim1=40*ones(1,symbols);
    lim2=60*ones(1,symbols);
    plot(lim1,'r');
    hold on
    plot(lim2,'r');
    title('Convergence plot for QPSK-Gardner technique');
    ylabel('tau axis'), xlabel('iterations')
    legend( ['SNRdB=' int2str(SNRdB(cv))]);
    axis([1 symbols 0 Tsym]);

    subplot(2,1,2);
    symbols = 2000;
    plot(remind(1:symbols),'*-');
    hold on
    lim1=40*ones(1,symbols);
    lim2=60*ones(1,symbols);
    plot(lim1,'r');
    hold on
    plot(lim2,'r');
    title('Convergence plot for QPSK-Gardner technique');
    ylabel('tau axis'), xlabel('iterations')
    legend( ['SNRdB=' int2str(SNRdB(cv))]);
    axis([1 symbols 0 Tsym]);

    %% Calculating the simulated BER
    Error=0;                    %Set the initial value for Error
    for k=1:N-1                 %Hard decision is taken

        if  (real(a(k))> 0 && real(data(k))==-1)||...
                (real(a(k))< 0 && real(data(k))==1)
            Error=Error+1;
        end
        if (imag(a(k))> 0 && imag(data(k))==-1)||...
                (imag(a(k))< 0 && imag(data(k))==1)
            Error=Error+1;
        end
    end
    BER_sim(cv)=Error/(2*(N-1));%Calculate error/bit
end
%% Plot BER Vs SNR
BER_th=qfunc(sqrt(2*SNR));      %Calculate The theoretical BER
figure
semilogy(SNRdB,BER_th,'b-','LineWidth',2); %Plot theoretical BER
hold on
semilogy(SNRdB,BER_sim,'r-','LineWidth',2);%Plot theoretical BER
```

```
title('SNR Vs. BER for QPSK- Gardner technique');
legend('Theoretical','Simulation');
ylabel('log BER');
xlabel('SNR in dB');
```

The above Matlab codes represent the baseline codes of the Gardner algorithm for QPSK. The following figures states the original convergence plots for QPSK-Gardner technique with SNR vs. BER plot:

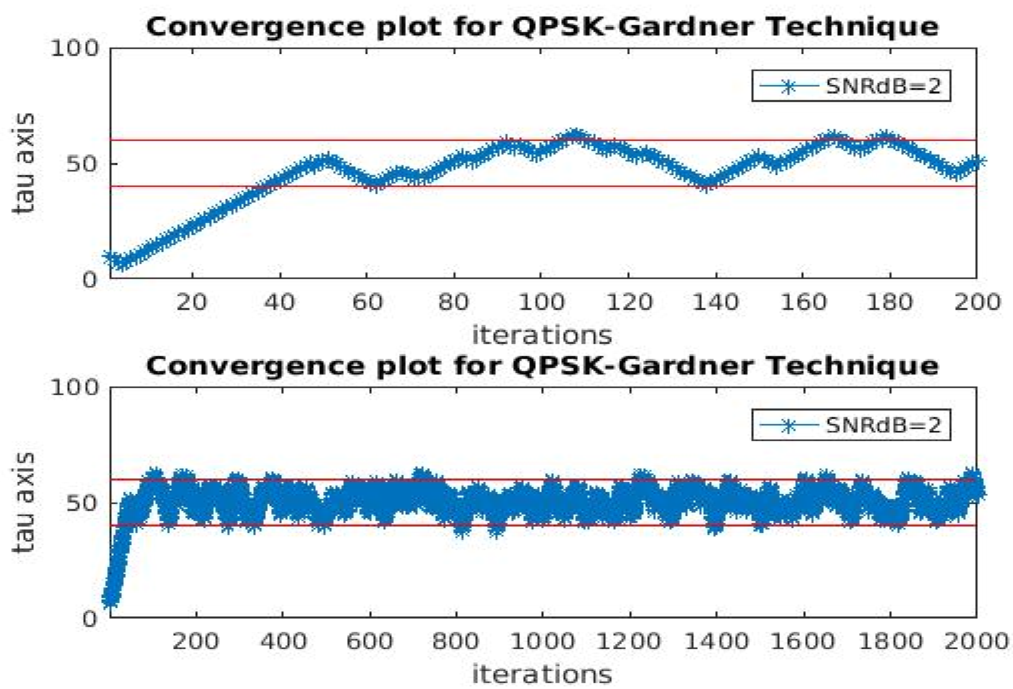

Figure 3.53: QPSK- Gardner technique, SNR=2 dB

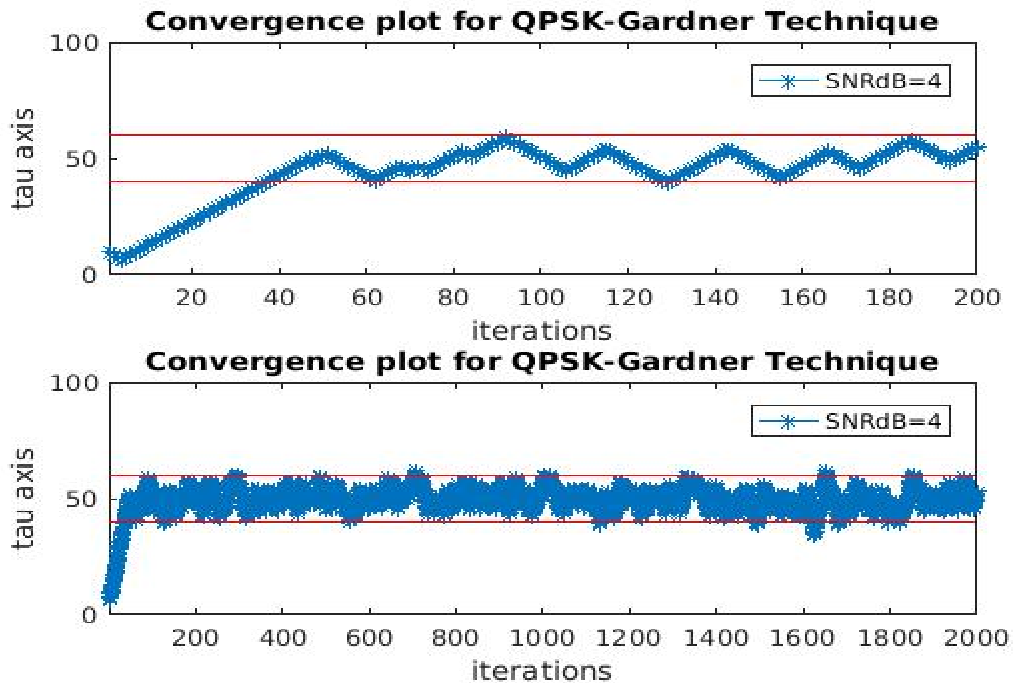

Figure 3.54: QPSK- Gardner technique, SNR=4 dB

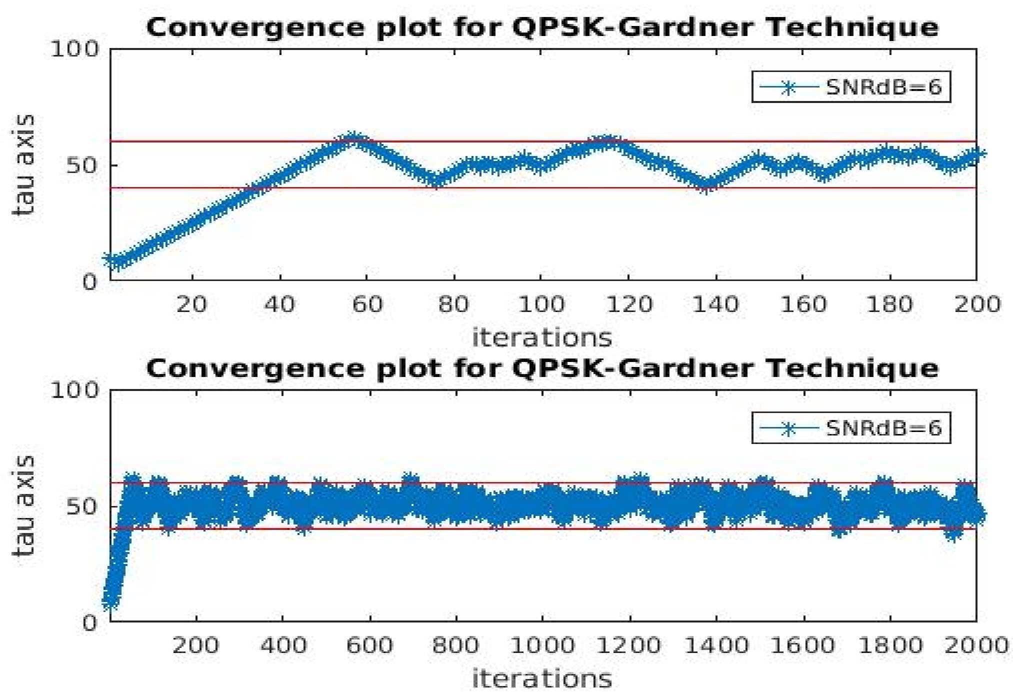

Figure 3.55: QPSK- Gardner technique, SNR=6 dB

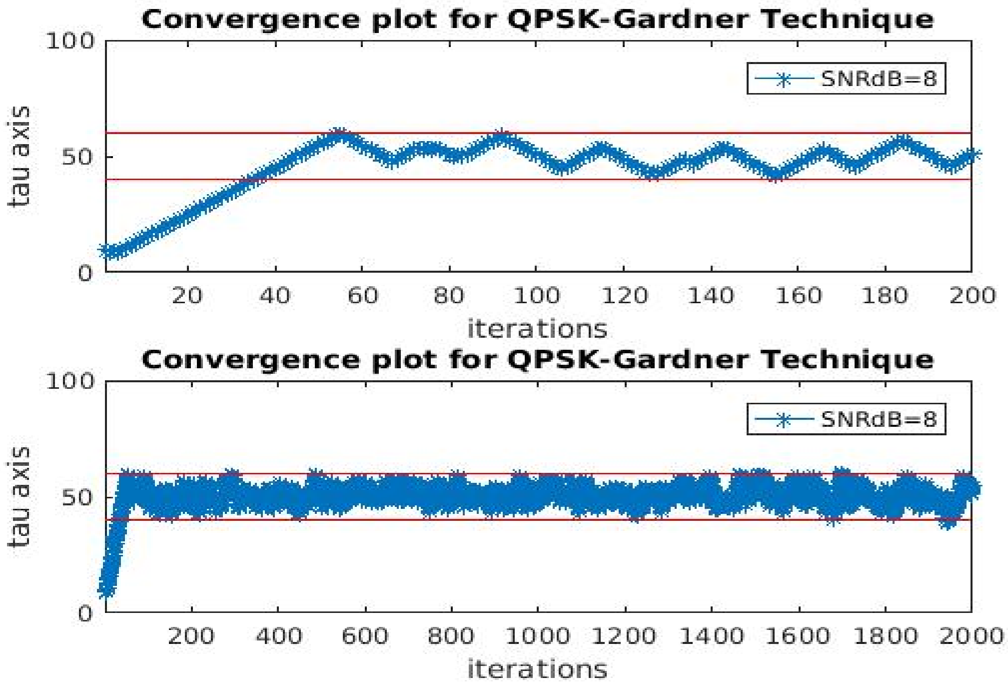

Figure 3.56: QPSK- Gardner technique, SNR=8 dB

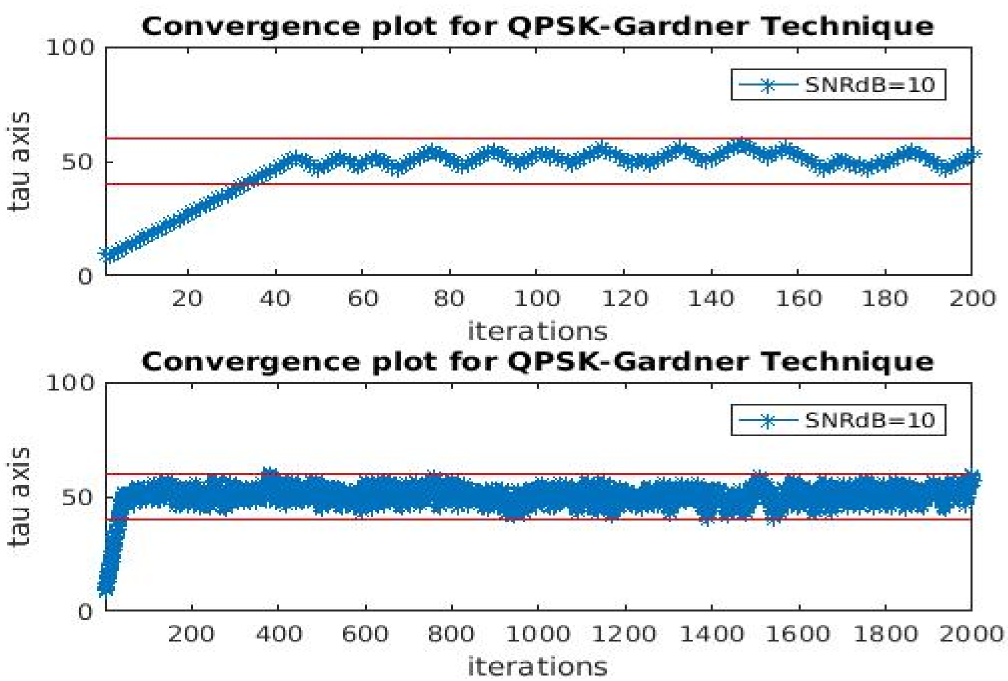

Figure 3.57: QPSK- Gardner technique, SNR=10 dB

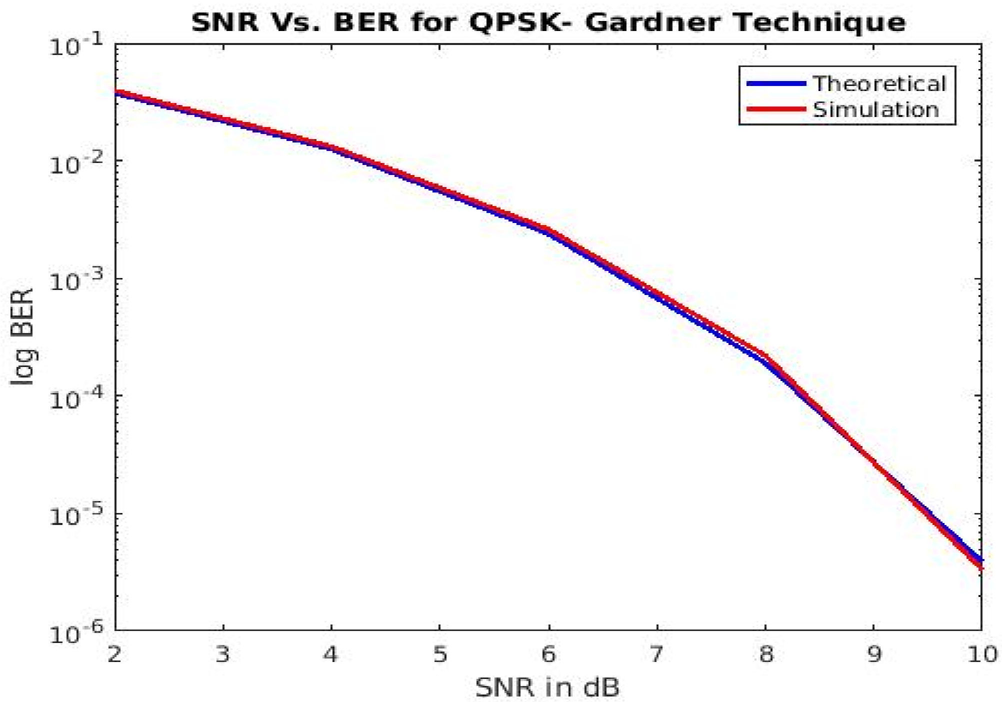

Figure 3.58: QPSK- Gardner technique, SNR vs. BER plot

Finally, the following table states the values of MSE that are corresponding to the SNR values for the original convergence plots.

SNR	2 dB	4 dB	6 dB	8 dB	10 dB
MSE	22.1758	21.9947	15.6789	12.7358	12.5737

Table 3.9: states SNR values with MSE value for QPSK-Gardner technique

Now, it is important to mention about the five techniques that are included in the following sections. Actually, the five techniques, which are used to improve timing recovery for BPSK-Gardner technique, are also used to improve the timing recovery for QPSK-Gardner technique. The results of the five techniques for QPSK state that they have the same features that are mentioned when the five techniques are used for BPSK. These features are in terms of convergence plots, SNR vs. BER plots, MSE tables, and the level of complexities of wireless communication systems. As a result, the explanation and the evaluation of each technique are not repeated again. So for more information, the five techniques that are used for BPSK modulation scheme can be reviewed again.

3.16 Summary and comparison of the five techniques - QPSK

It is worthwhile to summarize the five techniques that are introduced to work with QPSK modulation scheme. Moreover, the summary gives a clear picture about the benefits and features each technique. In order to facilitate the process of comparison, the convergence plots when SNR is equal to (10 dB) are be included. Furthermore, the tables of Mean Squared Error (MSE) are stated for each technique.

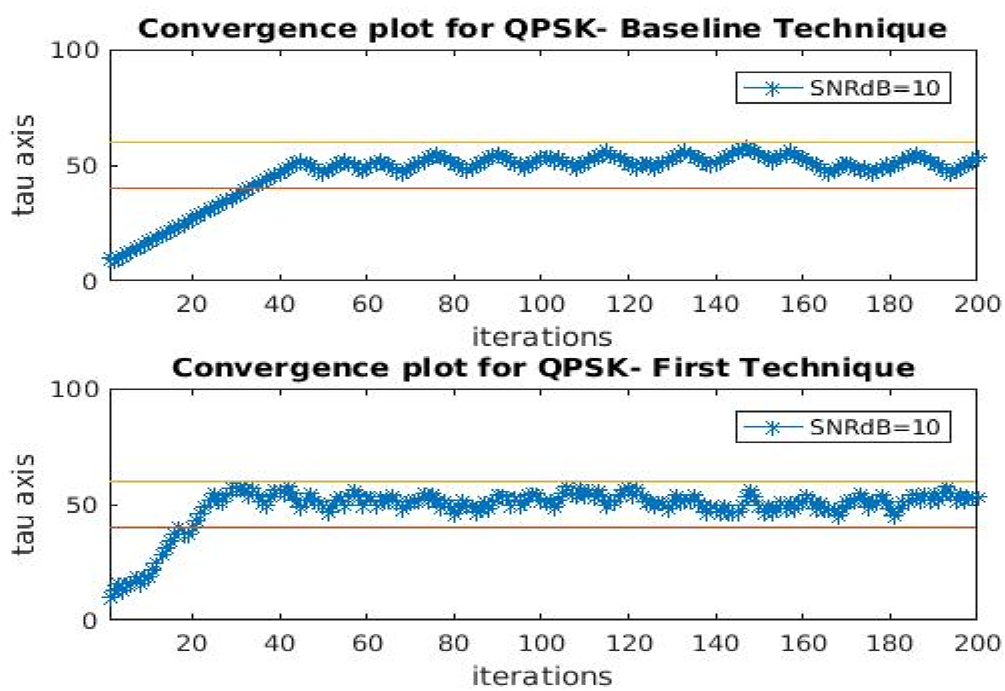

Figure 3.59: QPSK- Baseline technique and first technique, SNR=10 dB

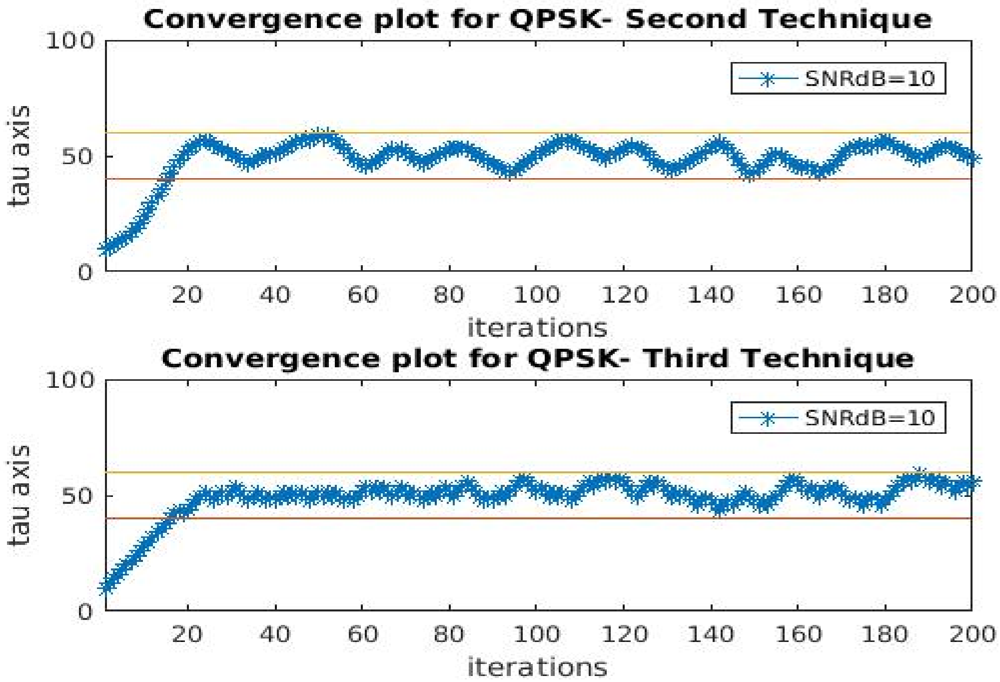

Figure 3.60: QPSK- second technique and third technique, SNR=10 dB

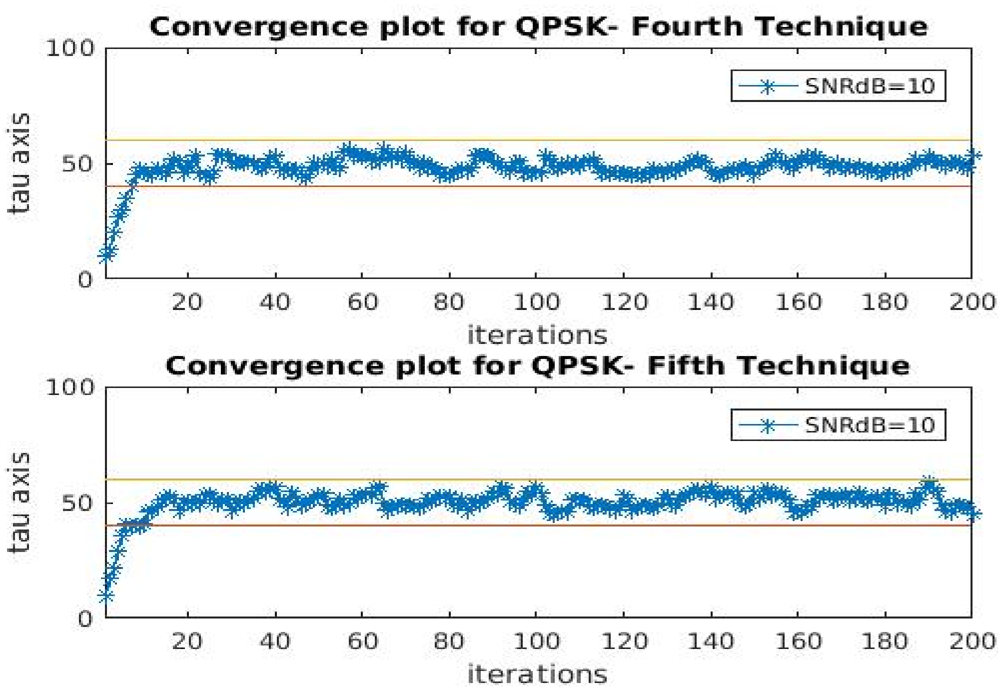

Figure 3.61: QPSK- fourth technique and fifth technique, SNR=10 dB

SNR	2 dB	4 dB	6 dB	8 dB	10 dB
MSE_Gardner technique	22.1758	21.9947	15.8474	12.7358	12.5737
MSE_first technique	58.5205	43.2153	29.9405	24.8537	17.6574
MSE_second technique	61.3295	44.8932	33.0179	19.9195	18.0674
MSE_third technique	27.3726	22.2947	20.9895	16.3242	13.5200
MSE_fourth technique	55.3126	35.4495	22.5968	15.4326	9.7863
MSE_fifth technique	34.9042	23.1337	16.7316	12.6284	8.5189

Table 3.10: Summary of MSE of all techniques for QPSK-Gardner technique

3.17 Alamouti technique with and without Gardner technique

In this section, the baseline Matlab code for the Alamouti technique is presented. Alamouti scheme includes two transmitters and a one receiver. In addition, the new Matlab code that allows for the Alamouti technique to work with the Gardner technique is introduced. As mention previously in the introduction, Alamouti [2] assumes that his technique works with digital receivers that are perfectly synchronized. In this thesis, The Alamouti technique is adapted with the Gardner technique to make the Alamouti technique works with digital receivers that are not perfectly synchronized. Consequently, this achieves a completely wireless system that works in realistic environment which has the Rayleigh fading and noise. In other words, this new technique reduces the effects of the Rayleigh fading by using the Alamouti technique; in addition, the new technique improves the timing recovery by implementing the five techniques that depend on the Gardner technique.

The QPSK data is generated, and it then splits into two streams. Then, the initial values for the Alamouti technique with the Gardner technique are provided. The (`h11` and `h22`) are generated to be used as a channel gain. Then, the (`h1` and `h2`) represent the effect of a slow fading which change every symbol in the data. Also, the noise is produced, and the noise changes every sample per symbol. Next, the pulse shape is applied on the each stream of the data. Next, the initial values that are used by the baseline code of the Alamouti technique is provided. The initial values include the channel gain (`hp1` and `hp2`). The noise is also generated to be used in the Alamouti algorithm. The following code demonstrates how to generate these initial values. The next Matlab code includes the data generation, initial values, and the pulse shape:

QPSKAlamouti_Gardner.m: Data generation, initial values, and the pulse shape.

```
%% Data generation
N=15*10^5;                          %Amount of data
Qpsk=[1+1i 1-1i -1+1i -1-1i];       %Four possibilities for QPSK
data= Qpsk(randi(4,1,N))./sqrt(2); %The data
Tsym=100;                           %No.of samples per symbol
xh1=data(1:2:end);                  %The first stream
xh2=data(2:2:end);                  %The second stream

%% Initial Parameter for Alamouti technique with Gardner technique
h11=(randn(1,N/2)+1i*randn(1,N/2))./sqrt(2);
                                    %Generating random numbers for h1
h22=(randn(1,N/2)+1i*randn(1,N/2))./sqrt(2);
                                    %Generating random numbers for h2
h1=kron(h11, ones(1,Tsym));         %The first channel gain
h2=kron(h22, ones(1,Tsym));         %The second channel gain
noise1=(randn(1,N*Tsym/2)+1i*randn(1,N*Tsym/2)./sqrt(2));
                                    %Generating random numbers for n1
noise2=(randn(1,N*Tsym/2)+1i*randn(1,N*Tsym/2)./sqrt(2));
                                    %Generating random numbers for n2
```

```
%% pulse shape
p = sin(2*pi*(0:Tsym-1)/(2*Tsym)); %Sinusoidal wave
data_up =zeros(1,length(xh1)*Tsym);%A memory of zeros
data_up(1:Tsym:end) = xh1;          %Interpolation the data
S11 = conv(data_up,p);              %The convolution operation
S1=S11(1:end-99);                   %Remove last 99 bits that are
                                    %added due to the convolution
%% pulse shape
p = sin(2*pi*(0:Tsym-1)/(2*Tsym));  %Sinusoidal wave
data_up = zeros(1,length(xh2)*Tsym);%A memory of zeros
data_up(1:Tsym:end) = xh2;           %Interpolation the data
S22 = conv(data_up,p);               %The convolution operation
S2=S22(1:end-99);                    %Remove last 99 bits that are
                                     %added due to the convolution
%% Initial Parameter for baseline Alamouti technique
hp1=(randn(1,N/2)+1i*randn(1,N/2))./sqrt(2);
                                     %The first channel gain
hp2=(randn(1,N/2)+1i*randn(1,N/2))./sqrt(2);
                                     %The second channel gain
noisep1=(randn(1,N/2)+1i*randn(1,N/2)./sqrt(2));
                                     %Generating random numbers
noisep2=(randn(1,N/2)+1i*randn(1,N/2)./sqrt(2));
                                     %Generating random numbers
```

After providing the initial values, the Signal to Noise (SNR) values are included, and the range of SNR is from (5 to 25 dB). The sigma value is found to be used in generating the noise (`np1` and `np2`) and to apply the Alamouti algorithm. Then, the noise is added to find the two received streams at the receiver (`rr1` and `rr2`). After that, the two combined streams are found (`S1_est` and `S2_est`) by using the two received streams.

After that, the two combined streams are recombined to create a one stream data (`xenlast`). Then, the simulated Bit Error Rate (BER) is calculated by comparing the real and imaginary parts of the combined stream with the real and imaginary parts of the original data respectively. The following Matlab code demonstrates the baseline Alamouti technique (part #1) and BER calculation:

QPSKAlamouti_Gardner.m: Baseline Alamouti technique and BER calculation.

```
SNRdB=5:5:25;                                 %Signal to Noise Ratio
SNR=10.^(SNRdB/10);                           %The linear values of noise
for cv=1:length(SNRdB)                        %Generate a loop
    %% Alamouti technique with perfect synchronization (part #1)
    sigma=sqrt(1)/sqrt(2*SNR(cv));            %Sigma generation
    np1=sigma*noisep1;                        %Generate the part#1 of noise
    np2=sigma*noisep2;                        %Generate the part#2 of noise
    rr1 = xh1.*hp1 + xh2.*hp2 + np1;          %The received signal at (t)
    rr2=-conj(xh2).*hp1+conj(xh1).*hp2+np2;
                                              %The received signal at (t+T)
    S1_est=conj(hp1).*rr1+hp2.*conj(rr2);
                                              %Stream#1 of combined signals
    S2_est=conj(hp2).*rr1-hp1.*conj(rr2);
                                              %Stream#2 of combined signals
    xenlast=zeros(1,N);                       %A memory for preallocating

    % Combining the two streams
    for m=1:N/2
        xenlast(2*m-1)=S1_est(m);             %#1 stream in odd order
        xenlast(2*m)=S2_est(m);               %#2 stream in even order
    end

    %% Calculating the simulated BER for part #1
    Error_PS=0;                               %Initial error for part #1
    for k=1:N                                 %Hard decision is used
       if  (real(xenlast(k))> 0 && real(data(k))==-1/sqrt(2))||...
                (real(xenlast(k))< 0 && real(data(k))==1/sqrt(2))
            Error_PS=Error_PS+1;
       end
       if  (imag(xenlast(k))> 0 && imag(data(k))==-1/sqrt(2))||...
                (imag(xenlast(k))< 0 && imag(data(k))==1/sqrt(2))
            Error_PS=Error_PS+1;
       end
    end
    BERps_sim(cv)=Error_PS/(N);               %Calculate errors/bits
```

The next step is how to find the BER in realistic environment that involves a digital receiver that is not perfectly synchronized. This step is the main contribution of this thesis. This code (part #2) starts with using values of noise `(n1` and `n2)` to find the two received streams `(r1` and `r2)`. Then, the two received streams are used to find the two combined streams (`S1g_est` and `S2g_est`). When the two combined streams are produced, they are fed to the Gardner technique. Each combined stream is represented by complex numbers, so each stream has real part and imaginary part. Consequently, there are four streams of bits that can be used in the timing error detection. After producing the four streams of bits, the initial values for the operation of error detection and correction are introduced. The following code connects the Alamouti technique with the Gardner technique and sets the initial values:

QPSKAlamouti_Gardner.m: Baseline Alamouti technique and BER calculation.

```
%% Alamouti technique with Gardner technique (part #2)
n1=sigma*noise1;            %Generate the first part of noise
n2=sigma*noise2;            %Generate the second part of noise
r1 = S1.*h1 + S2.*h2 + n1; %The received signal at (t)
r2=-conj(S2).*h1 + conj(S1).*h2 + n2;
                             %The received signal at (t+T)
S1g_est=conj(h1).*r1 + h2.*conj(r2);
                             %Stream#1 of combined signals
S2g_est=conj(h2).*r1 - h1.*conj(r2);
                             %Stream#2 of combined signals

%% Clock recovery_Gardner technique
% Feed the first stream of bits to Gardner technique
S1_real=zeros(1, N*Tsym/2); %A memory for preallocaing
S1_imag=zeros(1, N*Tsym/2); %A memory for preallocaing
for tr=1:N*Tsym/2
     S1_real(tr)= [real(S1g_est(tr))];%Real bits(In-phase channel)
     S1_imag(tr)= [imag(S1g_est(tr))];%Imaginary bits(Quadrature
                                      %channel)
end
```

```
% Feed the second stream of bits to Gardner technique
S2_real=zeros(1, N*Tsym/2);           %A memory for preallocaing
S2_imag=zeros(1, N*Tsym/2);           %A memory for preallocaing
for tr=1:N*Tsym/2
     S2_real(tr)= [real(S2g_est(tr))];%Real bits(In-phase channel)
     S2_imag(tr)= [imag(S2g_est(tr))];%Imaginary bits(Quadrature
                                       %channel)
end

%% Detection and correction
tau=0;                      %Initial value for tau
delta=Tsym/2;               %The shifting value before and after
                            %the midway sample
center=60;                  %The assumed order for the first
                            %midway sample
a1=zeros(1,N/2-1);          %A memory for preallocaing
a2=zeros(1,N/2-1);          %A memory for preallocaing
a3=zeros(1,N/2-1);          %A memory for preallocaing
a4=zeros(1,N/2-1);          %A memory for preallocaing
remind=zeros(1,N-1);        %A memory for preallocaing
avgsamples=6;               %Six values of Gardner algorithm are
                            %used to
%find the average
stepsize = 1;               % Correction step size
rit=0;                      % Iteration counter
GAp1 = zeros(1,avgsamples);%A memory for preallocaing
GAp2 = zeros(1,avgsamples);%A memory for preallocaing
ap1=zeros(1,N/2-1);         %A memory for preallocaing
ap2=zeros(1,N/2-1);         %A memory for preallocaing
tauvector=zeros(1,1900);    %A memory for preallocaing
uor=0;                      %A counter for the tau vector
```

After producing the initial values, a loop is created for sampling operation. This includes four main sections (two sections for each combined streams), and each combined stream is used to find error samples (`GAp1` and `GAp2`) which are used to find the average of the Gardner algorithm (`gardaverage`). Then, the loop filter is used to shift the samples depending on the value of tau. In addition to the loop filter, the next Matlab code shows how to save “remind’ values which are necessary to plot the convergence behavior. The code also illustrates how to find the tau vector that is important to find the Mean Squared Error (MSE). Moreover, the next code demonstrates how to achieve the correction operation.

QPSKAlamouti_Gardner.m: Sampling, error detection, loop filter, and correction.

```
for ii= (Tsym/2)+1:Tsym:length(S1_real)-(Tsym/2)+1
     rit=rit+1;                                   %A counter

     % Sampling the real part for stream#1
     midsample1=S1_real(center);                  %The midway sample
     latesample1=S1_real(center+delta);           %The late sample
     earlysample1=S1_real(center-delta);          %The early sample
     a1(rit)=earlysample1;                        %Save samples
     sub1=latesample1-earlysample1;               %Subtraction operation

     % Sampling the imaginary part for stream#1
     midsample2=S1_imag(center);                  %The midway sample
     latesample2=S1_imag(center+delta);           %The late sample
     earlysample2=S1_imag(center-delta);          %The early sample
     a2(rit)=earlysample2;                        %Save samples
     sub2=latesample2-earlysample2;               %Subtraction operation

     % Error detection for stream#1
     GAp1(mod(rit,avgsamples)+1)=sub1*midsample1+sub2*midsample2;
                                                   %Gardner Algorithm
     % Sampling the real part for stream#2
     midsample1p2=S2_real(center);                 %The midway sample
     latesample1p2=S2_real(center+delta);          %The late sample
     earlysample1p2=S2_real(center-delta);         %The early sample
     a3(rit)=earlysample1p2;                       %Save samples
     sub1p2=latesample1p2-earlysample1p2;          %Subtraction operation

     % Sampling the imaginary part for stream#2
     midsample2p2=S2_imag(center);                 %The midway sample
     latesample2p2=S2_imag(center+delta);          %The late sample
     earlysample2p2=S2_imag(center-delta);         %The early sample
     a4(rit)=earlysample2p2;                       %Save samples
     sub2p2=latesample2p2-earlysample2p2;          %Subtraction operation

     % Error detection for stream#2
     GAp2(mod(rit,avgsamples)+1)=sub1p2*midsample1p2+...
```

```
     sub2p2*midsample2p2;
                                        %Gardner Algorithm
    gardaverage=mean(GAp1+GAp2);        %Finding the mean

    % Loop filter
    if  gardaverage> 0
        tau = -stepsize;                %Shift by decreasing
    else
        tau =  stepsize;                %Shift by increasing
    end

    % Safe remind values
    remind(rit)=rem((center-Tsym/2),Tsym);%Save remind values to
                               %find convergence plots
    % tau vector
    if rit>=100 && rit<2000    %tau vector from 100 to
                               %2000 where the convergence happens
        uor=uor+1;
        tauvector(uor)= (remind(rit)- (Tsym/2)).^2;
    end                        %Difference between the
                               %estimated tau & the optimal tau
    % Correction
    center=center+Tsym+tau;    %Adding the tau value
    if center>=length(S1_real)-(Tsym/2)+1 %Break the loop when
                               %the midway sample reaches to 51
                               %samples before the last sample
        break;
    end
end
```

Next, the Mean Squared Error (MSE) is computed. After computing the MSE, bits of the first stream are combined, and the same thing happens to bits of the second stream. Then, the all bits are combined to have the final received bits that are used in the BER calculation. After that, the speed of convergence is plotted for each SNR value. Then, the simulated BER is calculated for the part #2 which involves using the Alamouti technique with the Gardner technique. Finally, the simulated BER baseline code (part #1) and the simulated BER of new Alamouti-Gardner technique (part #2) are plotted by using the next code:

QPSKAlamouti_Gardner.m: MSE, data combination, BER calculation (part #2), convergence plot, and BER plot.

```
% Mean Squared Error (MSE)
MSE(cv)=mean(tauvector);          %Finding Mean Squared Error
% Combine bits of stream#1
for f=1:N/2-1
      ap1(f)=[(a1(f))+(a2(f)*1i)];%Recombine complex numbers

end
% Combine bits of stream#2
for f=1:N/2-1
     ap2(f)=[(a3(f))+(a4(f)*1i)]; %Recombine complex numbers

end
% Combine all bits
for f=1:N/2-1
     a(2*f-1:2*f)=[ap1(f) ap2(f)];%Recombine complex numbers of
end                               %all bits
%% convergence plot
figure
symbols = 200;
subplot(2,1,1);
plot(remind(1:symbols),'*-');
hold on
lim1=40*ones(1,symbols);
lim2=60*ones(1,symbols);
plot(lim1,'r');
hold on
plot(lim2,'r');
title('Convergence plot for QPSK-Alamouti-Gardner');
ylabel('tau axis'), xlabel('iterations')
legend( ['SNRdB=' int2str(SNRdB(cv))]);
axis([1 symbols 0 Tsym]);
subplot(2,1,2);
symbols = 2000;
plot(remind(1:symbols),'*-');
hold on
lim1=40*ones(1,symbols);
lim2=60*ones(1,symbols);
plot(lim1,'r');
hold on
```

```
    plot(lim2,'r');
    title('Convergence plot for QPSK-Alamouti-Gardner');
    ylabel('tau axis'), xlabel('iterations')
    legend( ['SNRdB=' int2str(SNRdB(cv))]);
    axis([1 symbols 0 Tsym]);
    %% Calculating the simulated BER for part #2
    Error=0;                              %Initial error
    for k=1:N-2                           %Hard decision is used
        if  (real(a(k))> 0 && real(data(k))==-1/sqrt(2))||...
             (real(a(k))< 0 && real(data(k))==1/sqrt(2))
        Error=Error+1;
        end
        if  (imag(a(k))> 0 && imag(data(k))==-1/sqrt(2))||...
             (imag(a(k))< 0 && imag(data(k))==1/sqrt(2))
        Error=Error+1;
        end
    end
    BER_sim(cv)=Error/(N-4);              %Calculate errors/bits
end

%% Plot BER Vs SNR
figure
semilogy(SNRdB,BERps_sim,'b-','LineWidth',2);%Plot SNR Vs. BER
(part #1)
hold on
semilogy(SNRdB,BER_sim,'r-','LineWidth',2);  %Plot SNR Vs. BER
(part #2)
title('SNR Vs. BER plot for QPSK-Alamouti-Gardner');
legend('Baseline Alamouti','Alamouti-Gardner');
ylabel('log BER');
xlabel('SNR in dB');
```

The above Matlab codes represent the new baseline codes of the Alamouti technique and the Gardner technique which work with each other for the first time. The following figures states the convergence plots for QPSK- Alamouti - Gardner (QAG) technique with SNR vs. BER plot:

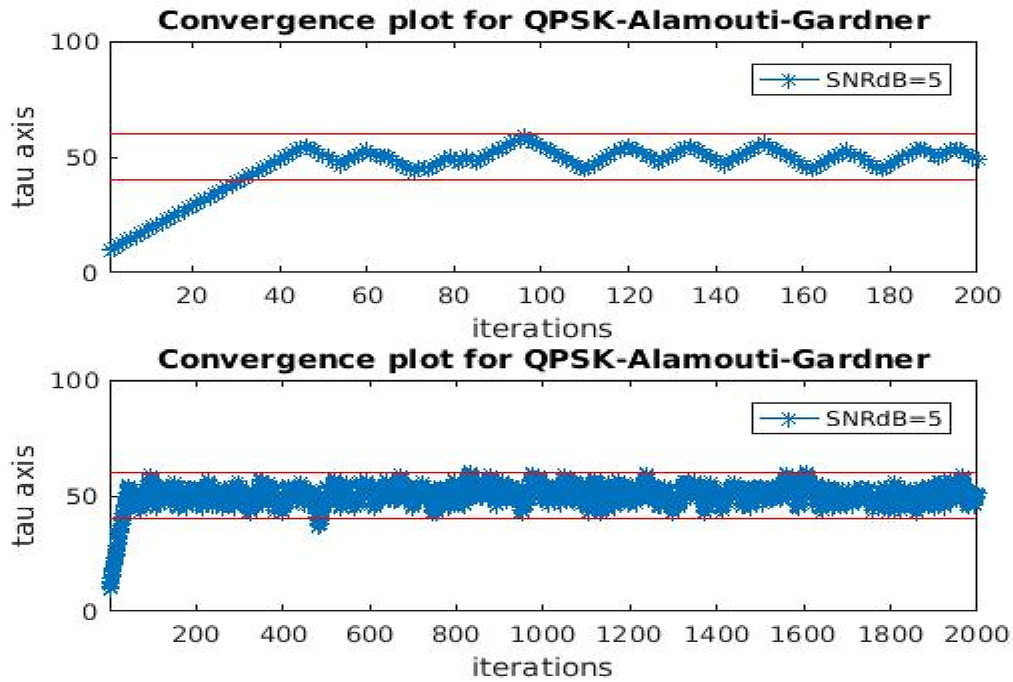

Figure 3.62: QPSK- Alamouti-Gardner technique, SNR=5 dB

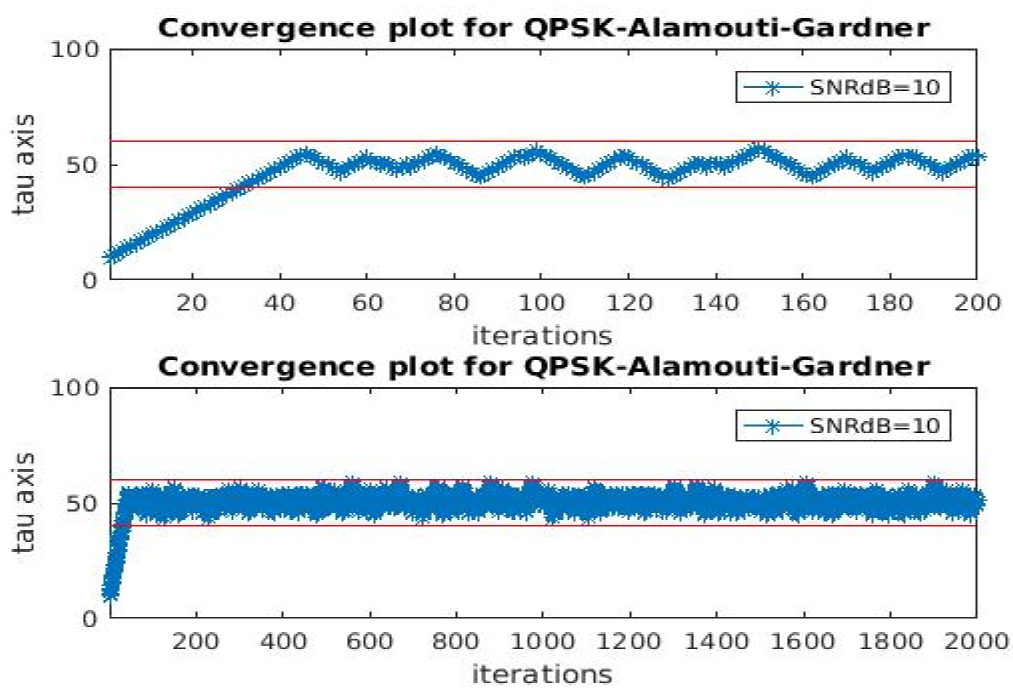

Figure 3.63: QPSK- Alamouti-Gardner technique, SNR=10 dB

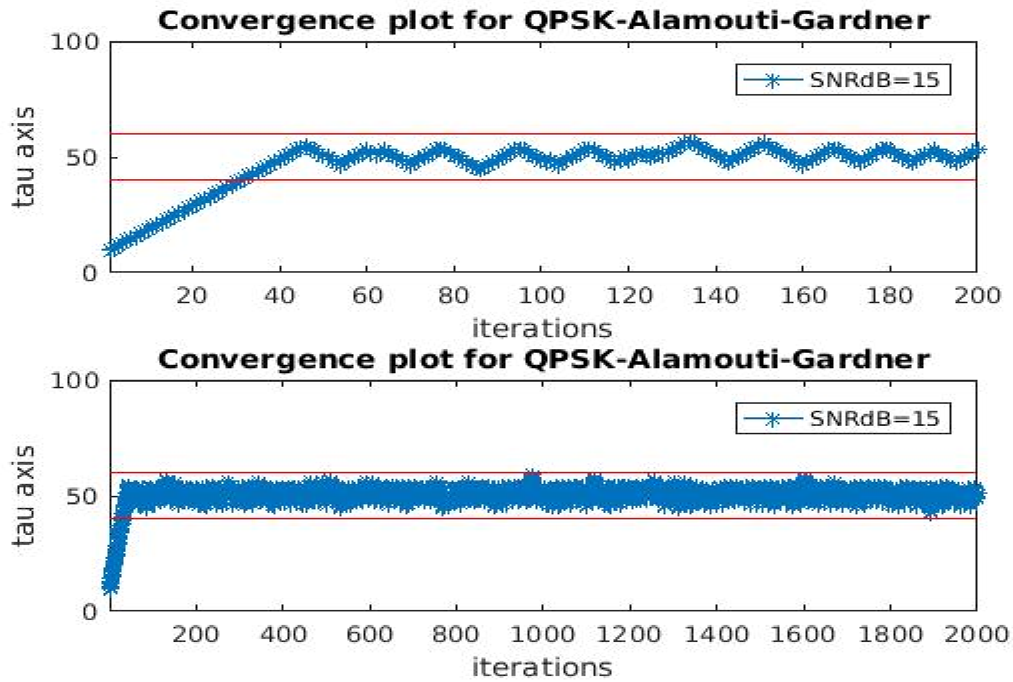

Figure 3.64: QPSK- Alamouti-Gardner technique, SNR=15 dB

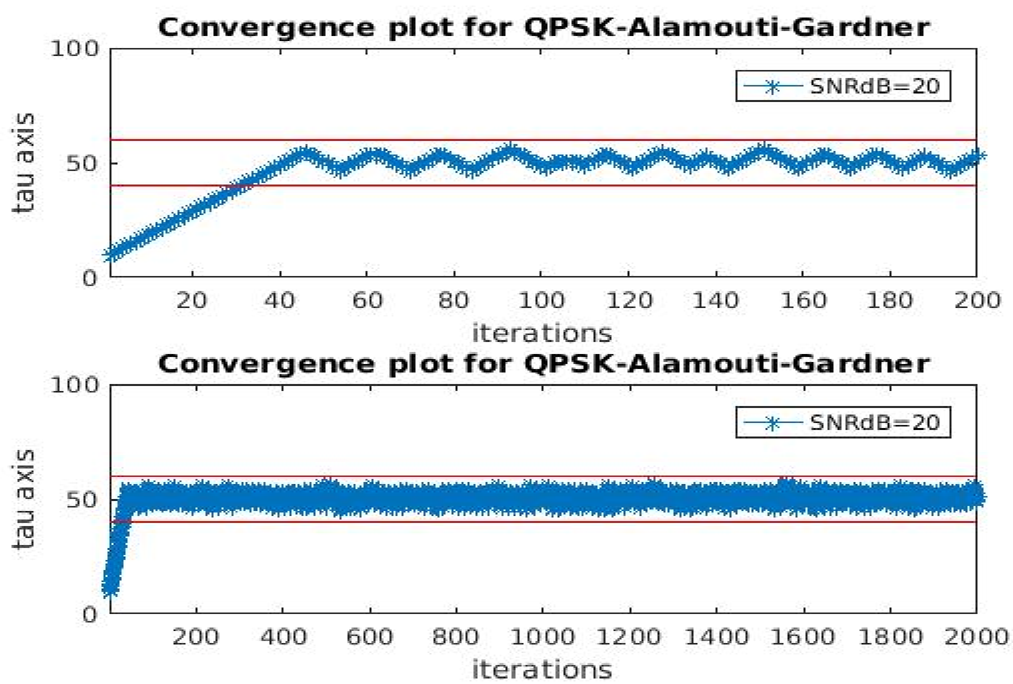

Figure 3.65: QPSK- Alamouti-Gardner technique, SNR=20 dB

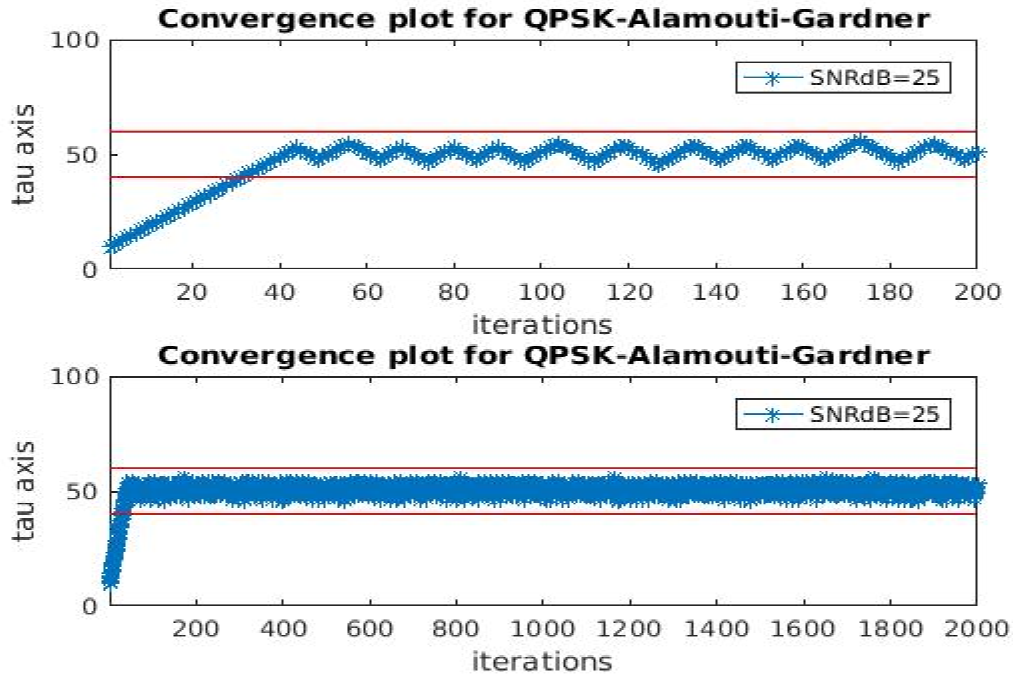

Figure 3.66: QPSK- Alamouti-Gardner technique, SNR=25 dB

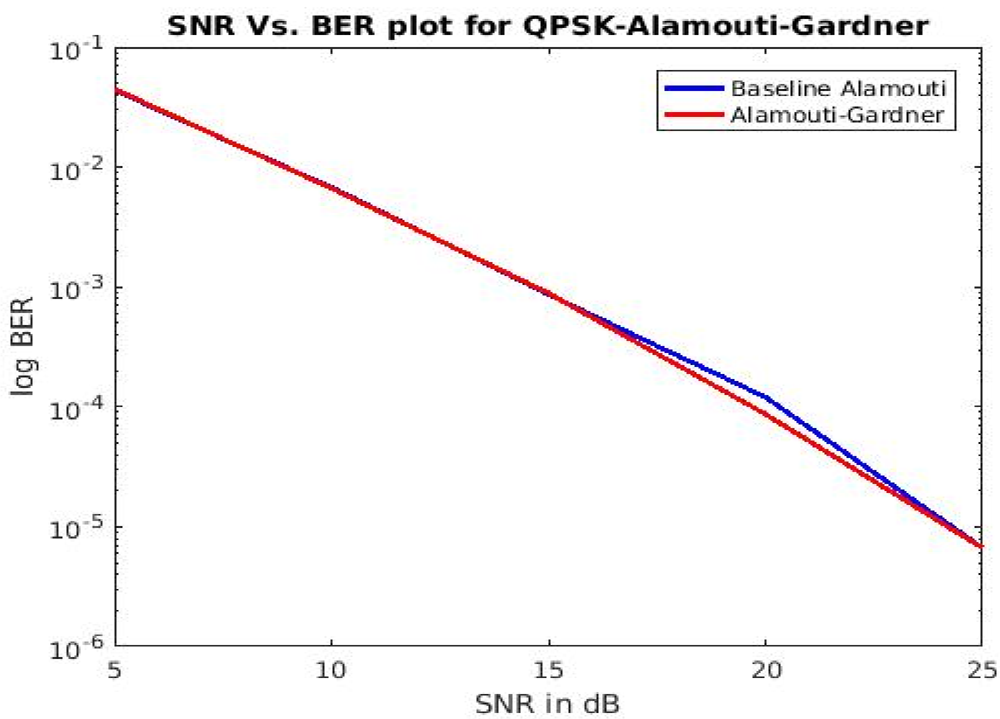

Figure 3.67: QPSK- Alamouti-Gardner technique, SNR vs. BER plot

The next table reveals the MSE values that are corresponding to the SNR values for the new QPSK- Alamouti - Gardner (QAG) technique.

SNR	5 dB	10 dB	15 dB	20 dB	25 dB
MSE	12.9716	7.7863	6.0137	5.1711	5.1016

Table 3.11: states MSE vs. SNR values for QPSK- Alamouti - Gardner (QAG) technique

The five techniques are used with QPSK - Alamouti - Gardner technique. The results illustrate that features and characteristics of these five techniques are the same as those that are mentioned in BPSK - Gardner technique. This thesis consider the QAG - third technique as the best technique because its features. These features include a faster convergence and less complexity in the design of wireless communication systems. The third technique also has the same SNR vs. BER plot that results from the QPSK - Alamouti - Gardner (QAG) technique. Additionally, the QAG - third technique has MSE values that are close to the MSE values of the QAG technique (see table 3.13). The following section explains the QAG - third technique and states its results.

3.18 The third technique to improve the QPSK-Alamouti- Gardner (QAG) Technique

The third technique uses the Matlab code of QPSK-Alamouti- Gardner (QAG) technique except the average filter code. This technique eliminates the average filter from the error detection operation, so the value of the Gardner algorithm is directly used in the loop filter. Also, the `step size`, which is used in this technique, is equal to 2. The following code demonstrates that:

QAG_thirdtechnique.m: Error detection and loop filter.

```
% Error detection for stream#1
GAp1=sub1*midsample1 + sub2*midsample2;         %Gardner Algorithm
% Error detection for stream#2
```

```
GAp2=sub1p2*midsample1p2 + sub2p2*midsample2p2;%Gardner Algorithm
gardaverage=GAp1+GAp2;                          %"gardaverage" finding
% Loop filter
if  gardaverage> 0
    tau = -stepsize;                            %Shift by decreasing
else
    tau=stepsize;                               %Shift by increasing
end
```

The convergence plots and SNR vs. BER plot are shown below:

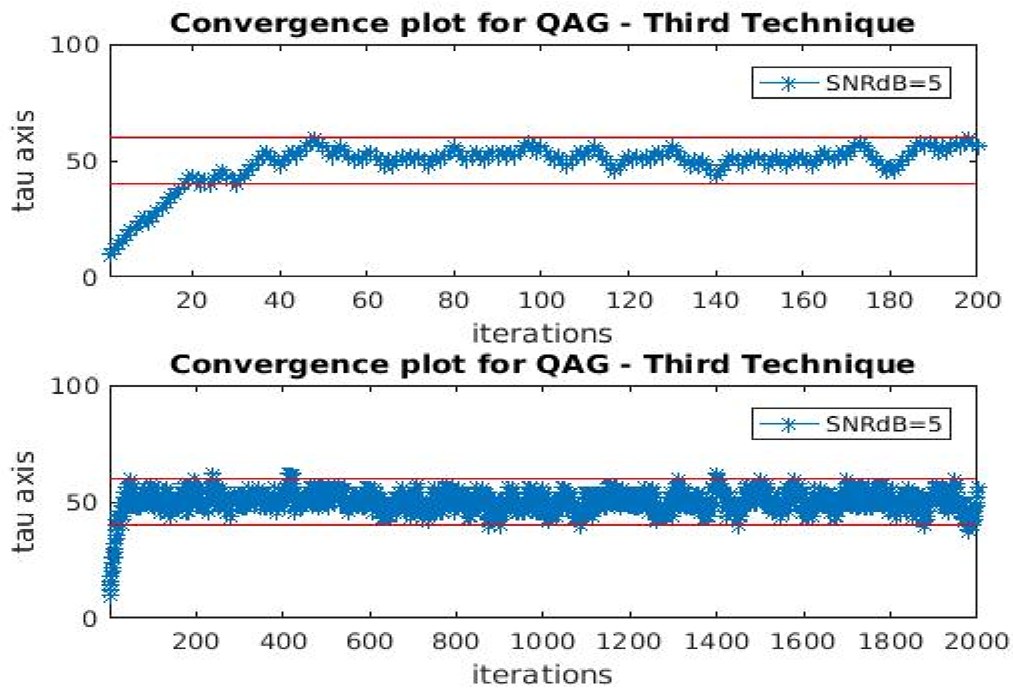

Figure 3.68: QAG - third technique, SNR=5 dB

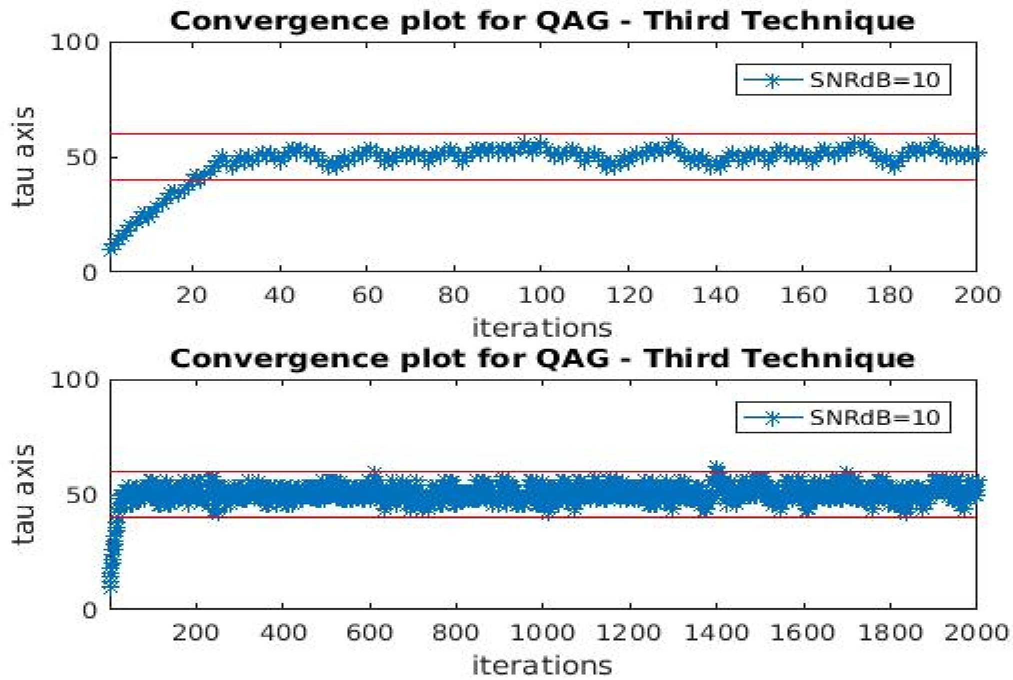

Figure 3.69: QAG - third technique, SNR=10 dB

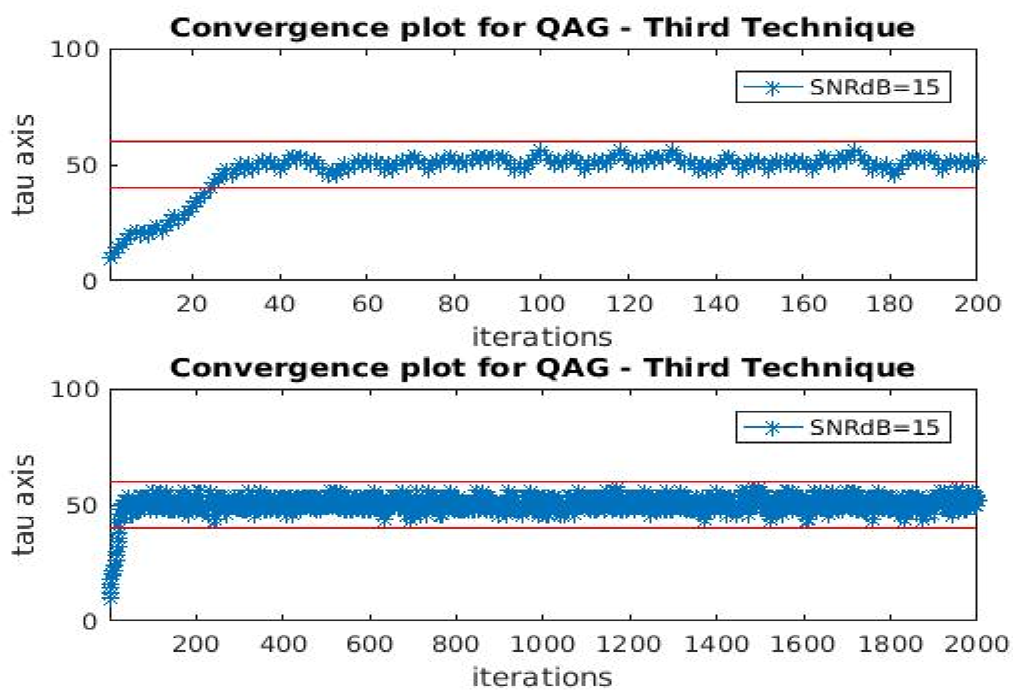

Figure 3.70: QAG - third technique, SNR=15 dB

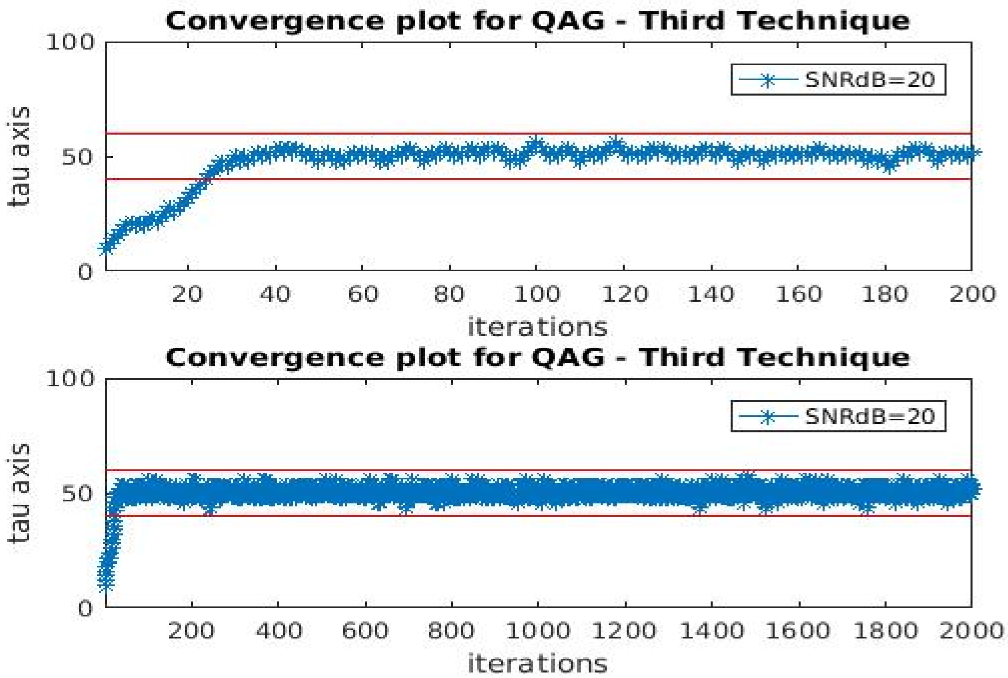

Figure 3.71: QAG - third technique, SNR=20 dB

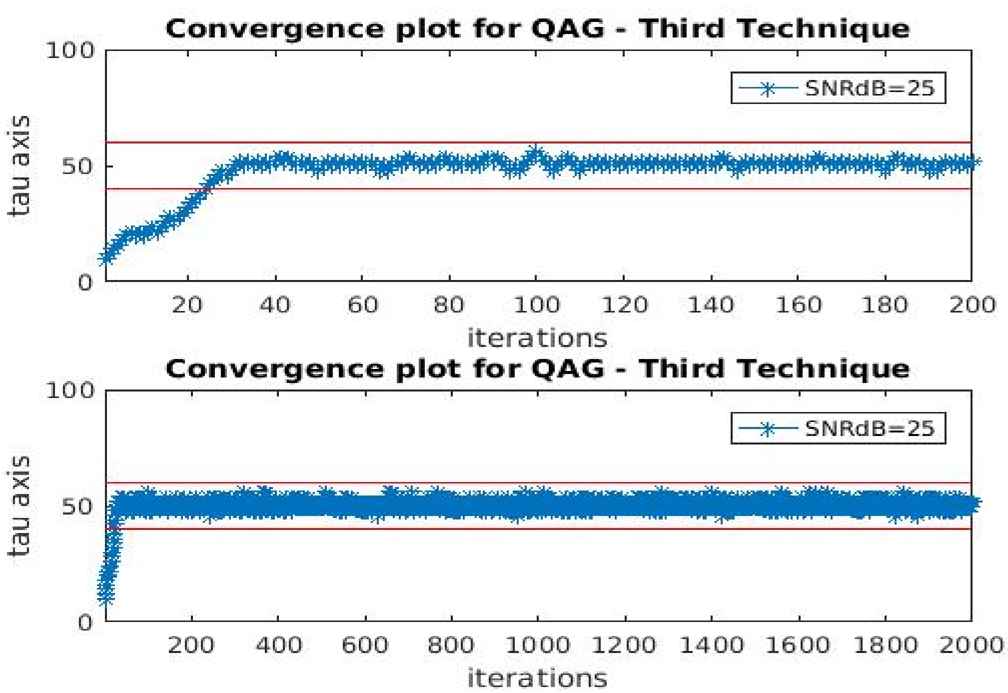

Figure 3.72: QAG - third technique, SNR=25 dB

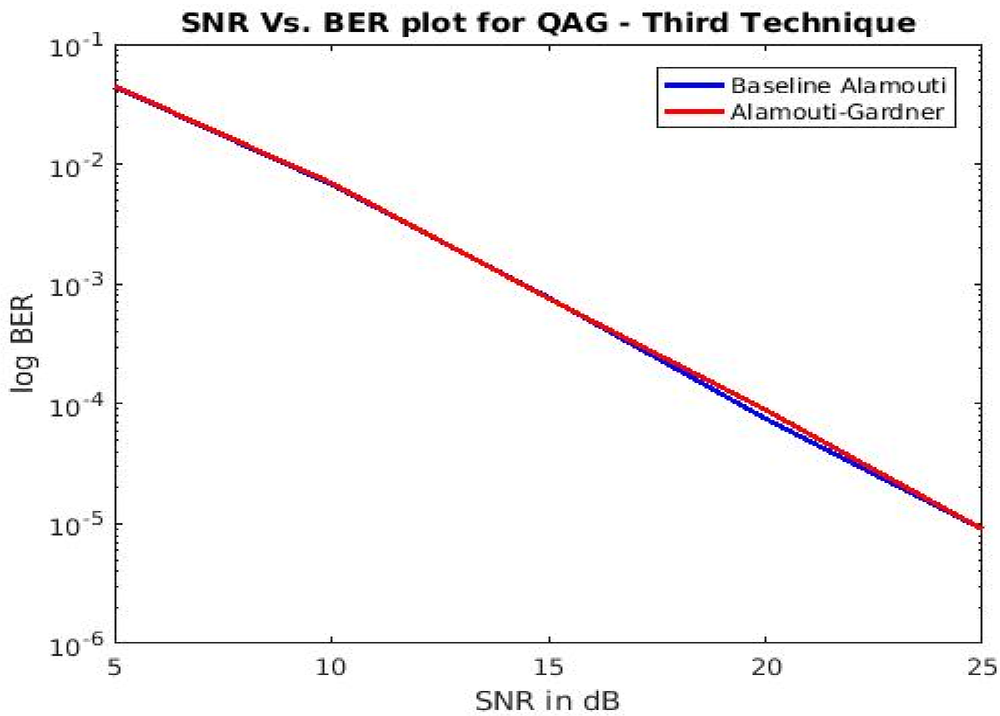

Figure 3.73: QAG - third technique, SNR vs. BER plot

The next table reveals the MSE values that are corresponding to the SNR values for the QPSK- Alamouti - Gardner (QAG) - third technique.

SNR	5 dB	10 dB	15 dB	20 dB	25 dB
MSE	14.4800	9.4526	6.4126	5.0232	3.7600

Table 3.12: states MSE vs. SNR values for QPSK- Alamouti -Gardner (QAG)- third technique

3.19 Summary and comparison of the five techniques that work with QPSK-Alamouti -Gardner (QAG) Technique

It is worthwhile to summarize the five techniques that are introduced to work with QPSK-Alamouti -Gardner (QAG) technique. Moreover, the summary gives a clear picture about the benefits and features each technique. In order to facilitate the process of comparison, the convergence plots, when SNR is equal to (25 dB), are included. Furthermore, the tables of Mean Squared Error (MSE) are stated for each technique.

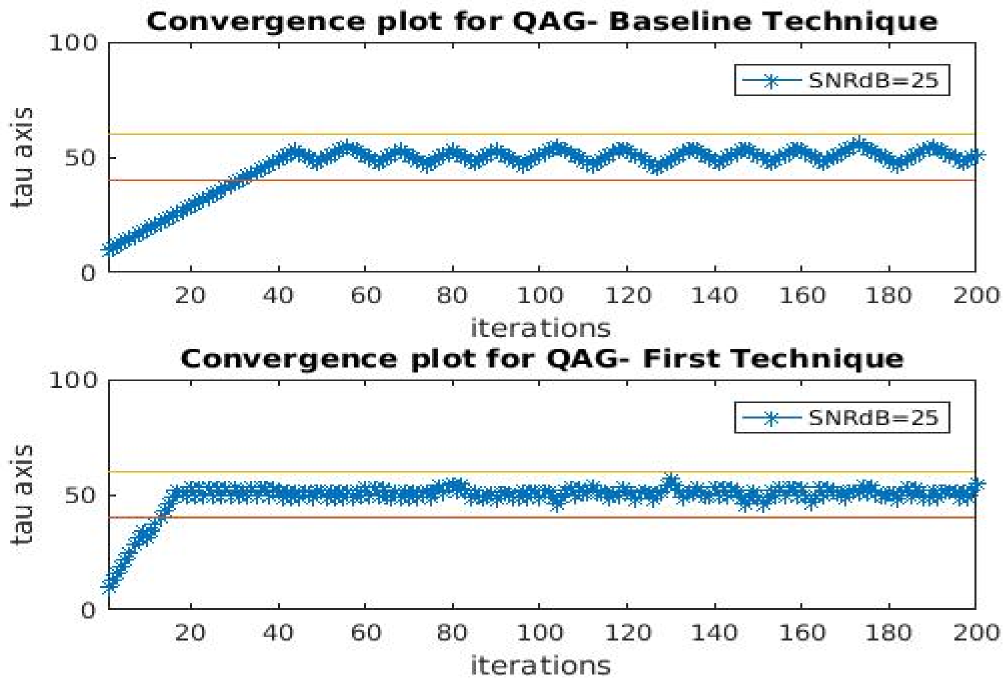

Figure 3.74: QAG -Baseline technique and first technique , SNR=25 dB

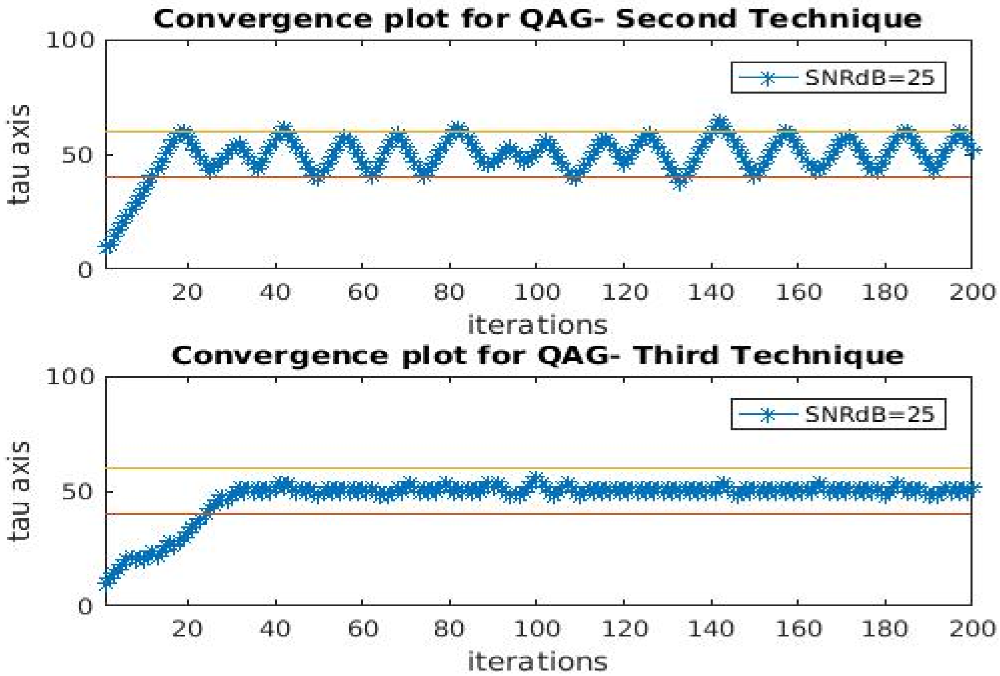

Figure 3.75: QAG- second technique and third technique, SNR=25 dB

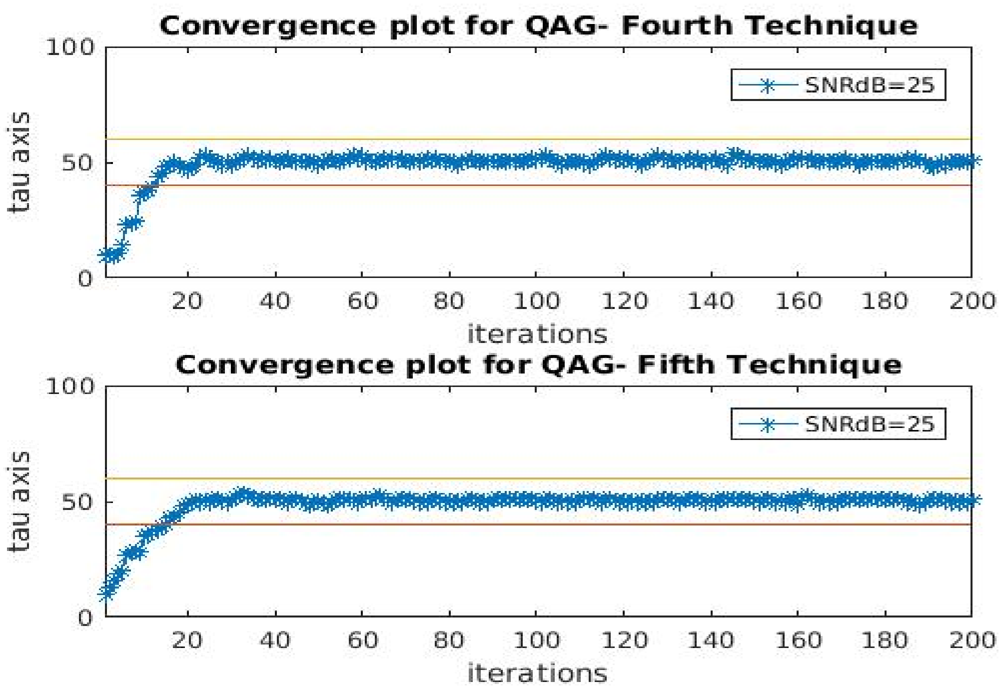

Figure 3.76: QAG- fourth technique and fifth technique, SNR=25 dB

SNR	5 dB	10 dB	15 dB	20 dB	25 dB
QAG technique	12.9716	7.7863	6.0137	5.1711	5.1016
QAG first technique	21.0232	12.7121	7.7305	6.0637	4.2716
QAG second technique	45.7153	36.9111	35.3679	34.0842	33.8011
QAG third technique	14.4800	9.4526	6.4126	5.0232	3.7600
QAG fourth technique	12.2179	5.6453	3.8116	2.7189	2.2516
QAG fifth technique	7.2853	4.0642	3.7274	2.4884	2.0600

Table 3.13: Summary of MSE of all techniques with QAG technique

The analysis of the above results states that all five techniques have a faster convergence. Moreover, all the five techniques have the same SNR vs. BER plot, so they have the same BER behavior. In term of the MSE values, each technique has different MSE values. However, behavior of the BER proves that these MSE values do not affect the performance of the five techniques. In term of the complexity of the wireless receiver design, each technique also has its own complexity because it depends on the use of filters in the structure. Generally, the third technique presents good solutions which include increasing in convergence speed, reducing the complexity of the wireless receiver design with having the same BER behavior and reasonable MSE values.

Finally, analyzing the results state that the first hypothesis has been realized which states that “the new algorithms improve the digital communication system performance in term of the convergence speed with reducing the complexities of the communication system design”.

3.20 Limitations

One of the important things that should be taken into consideration in wireless communication system is data rates. When high data rates are used in a wireless communication system, this allows for the researcher to get a clear picture and accurate results. However, the high data rates require a high speed in the processing to avoid being late. As a result, to increase the speed of processing, this requires a big memory and a high CPU speed in the lab computer. These big memory and high CPU speed are usually expensive.

The other limitation is the noise that is produced from external and internal environment which is not known. The external noise is the noise that comes from outside sources such as atmospheric noise, extraterrestrial noise, and industrial noise. The internal noise is the noise that is generated within communication systems such as thermal noise. Because these kinds of noise are not known, it is hard to estimate their impacts in the proposed algorithms.

Chapter 4

Conclusion and Future Research

4.1 Conclusion

The importance Symbol synchronization techniques have increased due to the increased demand on the bandwidth and the quality of services. Wireless communication systems face difficulties due to the increased noise and Rayleigh fading. The additional components, which are added to improve wireless systems, may increase the processing time and computational complexities of wireless communication systems. In addition, using wireless networks are growing rapidly which increases the problem of multipath fading.

The Alamouti technique is used to reduce the Rayleigh fading effects in the digital communication systems. Moreover, this thesis uses Quadrature Phase Shift Keying (QPSK) which has the ability to transmit high data rates. Furthermore, the Gardner technique is a symbol synchronization technique that is improved by this thesis. This thesis introduces new techniques to improve the performance of symbol synchronization by reducing complexities in wireless receiver design and by increasing the convergence speed with having the same BER measurements and reasonable Mean Squared Error (MSE) values.

In this thesis, the Alamouti space-time code technique is written for QPSK modulation scheme to work in realistic environment that involves a timing synchronization technique. We compare the bit error rate (BER) of the Alamouti decoder when synchronized using the proposed algorithms with the ideal results found in the literature, and we find them to be similar, proving that the synchronization algorithm is in fact achieving optimum synchronization.

4.2 Future Research

The modulation schemes that are used in this thesis are BPSK and QPSK. So, it will be a good idea to use different modulation schemes to evaluate the performance of the new five techniques. In addition, the channel conditions that are involved in this thesis include Additive white Gaussian noise (AWGN) and Rayleigh fading. Consequently, other types of fading can be taken into the consideration to analysis the performance of the five techniques.

For the timing correction, this thesis assumes simulate the impact of the interpolator by assuming that there are 100 samples per symbol. So, other interpolation techniques can be implemented for the timing recovery. Another scenario of Alamouti technique can be implemented by using two transmitters and two receivers. Furthermore, hardware implementation can be accomplished to obtain results and verify them with the simulation results in terms of convergence speeds, MSE values, BER measurements, and the level of complexities in wireless communication systems.

Bibliography

[1] F. Gardner. A BPSK/QPSK timing-error detector for sampled receivers. *IEEE Transactions on Communications 34(5),* pp. 423-429. 1986. DOI: 10.1109/TCOM.1986.1096561.

[2] S. M. Alamouti. A simple transmit diversity technique for wireless communications. *IEEE Journal on Selected Areas in Communications 16(8),* pp. 14511458. 1998. DOI: 10.1109/49.730453.

[3] M. Kihara *et al. Digital Clocks for Synchronization and Communications* 2003.

[4] J. P. Costas. Synchronous communications. *Proceedings of the IRE 44(12),* pp. 1713-1718. 1956. DOI: 10.1109/JRPROC.1956.275063.

[5] L. Franks. Carrier and bit synchronization in data communication--A tutorial review. *IEEE Transactions on Communications 28(8),* pp. 1107-1121. 1980. DOI: 10.1109/TCOM.1980.1094775.

[6] C. R. Johnson and W. A. Sethares, *Telecommunication breakdown: concepts of communication transmitted via software-defined radio*. Upper Saddle River, NJ: Pearson Education Inc., 2004.

[7] A. Goldsmith. Wireless communications. *Cambridge University Press.* 2005.

[8] H. Takahashi *et al*. 120-GHz-band fully integrated wireless link using QSPK for realtime 10-Gbit/s transmission. *IEEE Transactions on Microwave Theory and Techniques 61(12),* pp. 4745-4753. 2013. DOI: 10.1109/TMTT.2013.2285354.

[9] P. R. Strode and P. Groves. GNSS multipath detection using three-frequency signal-to-noise measurements. *GPS Solutions 20(3),* pp. 399-412. 2016. DOI: 10.1007/s10291-015-0449-1.

[10] K. Mueller and M. Muller. Timing recovery in digital synchronous data receivers. *IEEE Transactions on Communications 24(5),* pp. 516-531. 1976. DOI: 10.1109/TCOM.1976.1093326.

[11] T. Suzuki *et al*. Line equalizer for a digital subscriber loop employing switched capacitor technology. *IEEE Transactions on Communications 30(9),* pp. 2074-2082. 1982. DOI: 10.1109/TCOM.1982.1095697.

[12] O. Agazzi *et al*. Timing recovery in digital subscriber loops. *IEEE Transactions on Communications 33(6),* pp. 558-569. 1985. DOI: 10.1109/TCOM.1985.1096341.

[13] A. Yazgan and I. H. Cavdar. Optimum link distance determination for a constant signal to noise ratio in M-ary PSK modulated coherent optical OFDM systems. *Telecommunication Systems 55(4),* pp. 461-470. 2014. DOI: 10.1007/s11235-013-9801-3.

[14] W. C. Lindsey and M. K. Simon, *Telecommunication Systems Engineering*. Englewood Cliffs, NJ: Prentice-Hall, 1973, ch. 9.

[15] E. L. Lehmann, G. Casella and I. ebrary. *Theory of Point Estimation* (2nd ed.) 1998.