**SIMULATION AND ANALYSIS OF SPECTRUM SENSING IN COGNITIVE RADIO**

A Project Report

Submitted in partial fulfillment of the requirement for the award of the degree of

**BACHELOR OF TECHNOLOGY**

In

**ELECTRONICS AND COMMUNICATION ENGINEERING**

Submitted by

**K. SWARNA LATHA 17191A0423**

**Y. NARENDRA REDDY 17191A0459**

**G. MUNI VENKATESH 17191A0415**

**T. VASAIAH 17191A0454**

**Under the Esteemed guidance of**

**Mr. P.G VARNA KUMAR REDDY M Tech, (Ph.D.),**

Assistant Professor (ADHOC), Department of ECE

JNTUACE, Pulivendula



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR COLLEGE OF ENGINEERING (AUTONOMUS)::PULIVENDULA PULIVENDULA-516390, Y.S.R(District), ANDHRA PRADESH, INDIA::2017-2021**

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR**

**COLLEGE OF ENGINEERING (AUTONOMOUS), PULIVENDULA**

**Y.S.R(District), ANDHRA PRADESH, INDIA**

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**



**CERTIFICATE**

This is to certify that the project work entitled **“SIMULATION AND ANALYSIS OF SPECTRUM SENSING IN COGNITIVE RADIO”** has been submitted by

**K. SWARNA LATHA 17191A0423**

**Y. NARENDRA REDDY 17191A0459**

**G. MUNI VENKATESH 17191A0415**

**T. VASAIAH 17191A0454**

in partial fulfillment of the requirements for the award of the **Bachelor of Technology** in **Electronics and Communication Engineering** from **JNTUA College of Engineering (Autonomous), Pulivendula** during 2017-2021.

**Project Guide Head of the Department**

Mr. P. G VARNA KUMAR REDDY Dr. R. RAMANA REDDY

Assistant Professor (ADHOC), Professor and Head,

Department of ECE, Department of ECE,

JNTUACE, Pulivendula. JNTUACE, Pulivendula.

**Signature of project guide Signature of HOD**

**JAWAHARLAL NEHRU TECHNOLOGICALUNIVERSITY ANANTAPUR**

**COLLEGE OF ENGINEERING (AUTONOMOUS), PULIVENDULA**

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**



**STUDENT DECLARATION**

We hereby declare that the project entitled “**SIMULATION AND ANALYSIS OF SPECTRUM SENSING IN COGNITIVE RADIO”** is our own work and that to the best of our knowledge and belief, it contains no material previously published or material which has been accepted for the award of any degree or diploma of any University or institute of higher learning.

**K. SWARNA LATHA 17191A0423**

**Y. NARENDRA REDDY 17191A0459**

**G. MUNI VENKATESH 17191A0415**

**T. VASAIAH 17191A0454**

**ACKNOWLEDGEMENT**

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that we have now the opportunity to express our gratitude for all of them.

The first person we would like to thank is our guide **Mr. P.G Varna Kumar Reddy** Assistant Professor, Department of Electronics and Communication Engineering, J.N.T.U.A college of Engineering, Pulivendula. Her wide knowledge and logical way of thinking have made a deep impression on us. She is a source of inspiration for innovative ideas and her kind support is well known to all her students and colleagues.

We are grateful **to Dr. R. Ramana Reddy**, Professor and Head of the Department of Electronics and Communication Engineering, J.N.T.U.A college of Engineering, Pulivendula who has extended her support for the success of this project.

We wish to thank **Prof. G. Sankara Sekhar Raju,** Principal of the college, JNTUA college of Engineering, Pulivendula who has extended his support for the success of this project.

Finally, we would like to extend our deep sense of gratitude to all the staff members, friends and last but not least we are greatly indebted to our parents who inspired us at all circumstances.

**ABSTRACT**

To satisfy future bandwidth demands, existing Cognitive Radio with Wireless Communication must be upgraded to make the best use of the bandwidth. The primary objective of IEEE 802.22 standard is to determine vacant spectrum bands available in Digital television channel (DTV) and to utilize them for wireless rural broad band connectivity. Cognitive Radio (CR) aims at maximizing the utilization of the limited radio bandwidth while accommodating the increasing number of services and applications in wireless networks. For cognitive radio networks to operate efficiently. Secondary users (SU) should be able to exploit radio spectrum that is unused by the primary network. A critical component of cognitive radio is spectrum sensing.The solution for this problem is to use the cognitive radio technology. Cognitive Radio is a new paradigm in wireless communication to tackle the problem of spectrum underutilization. Cognitive radio technology gives the ability to use these licensed bands by means of spectrum sensing techniques and another technique.

There are many spectrum sensing algorithms available in the literature out of which energy detection is widely used because it is easy to implement and it does not require prior information about PU (Primary User). However, the performance of the conventional energy detector deteriorates in low SNR region. Double threshold CSS (Cooperative Spectrum Sensing) was introduced to increase the reliability of decision but at the cost of some sensing information lost. A method of double threshold CSS in which each CR (Cognitive Radio) sends local decision or observed energy to the FC (Fusion Center) depending on the region in which the observed energy lies. FC then makes a final decision by combining local decisions and observed energy values.

A simulation methodology is proposed for spectrum sensing technique to meet the requirements of IEEE 802.22 standard. This Project describes several simulation scenarios that can be used to evaluate spectrum sensing by single unit (local sensing) and multiple SU's (collaboratively). The detection performance is described through extensive simulation using MATLAB simulation tool.

**CONTENTS**

**S. no Topic Page no.**

I ABSTRACT

II LIST OF FIGURES & TABLES

**CHAPTER-1 INTRODUCTION (1-9)**

1.1 Software Defined Radio (SDR) 2

1.2 Cognitive Radio (CR) 2

1.3 Functions of Cognitive Radio 5

1.3.1 Spectrum Sensing 5

1.3.1.1 Non-Cooperative Spectrum Sensing technique 6

1.3.1.2 Cooperative Spectrum Sensing technique 6

1.3.2 Spectrum management 6

1.3.3 Spectrum Sharing 7

1.3.4 Spectrum Mobility 7

1.4 Types of Cognitive Radio 7

1.5 Characteristics of Cognitive Radio 7

1.6 Monte Carlo Simulation Technique 8

**CHAPTER-2 SOFTWARE DESCRIPTION (10-20)**

2.1 Introduction to MATLAB 10

2.2 Basic building blocks of MATLAB 11

2.2.1 MATLAB Window 11

2.2.2 Command Window 11

2.2.3 Workspace Window 11

2.2.4 Current Directory Window 12

2.2.5 Command History Window 12

2.2.6 Editor Window 12

2.2.7 Graphics or Figure Window 12

2.2.8 Online Help Window 12

2.3 The MATLAB System 13

2.3.1 Development Environment 13

2.3.2 The MATLAB Mathematical Function 13

2.3.3 The MATLAB Language 13

2.3.4 Graphics 13

2.3.5 The MATLAB Application Program Interface (API) 13

2.4 MATLAB Files 14

2.4.1 M-Files 14

2.4.1.1 Script Files 14

2.4.1.2 Function Files 14

2.5 MAT-Files 14

2.6 MATLAB Working Environments 14

2.6.1 MATLAB Desktop 14

2.6.2 Using the MATLAB Editor to create M-Files 15

2.7 Basic Commands 15

2.8 Some Basic Plot Commands 17

2.8.1 Kinds of Plots 17

2.8.2 Plot Description 18

2.9 Algebraic Operations in MATLAB 18

2.9.1 Scalar Calculations 18

2.10 Introduction to SIMULINK 19

2.11 Getting Help 20

**CHAPTER-3 SPECTRUM ALLOCATION BY USING**

**MATLAB AND SIMULINK (21-27)**

3.1 MATLAB 21

3.1.1 Spectrum Holes 21

3.1.2 Assigning Primary User to the Frequency Spectrum 21

3.1.3 Assigning new user to the Spectrum Holes 23

3.1.4 Efficient Frequency Bandwidth 24

3.1.5 Elimination of a Slot 24

3.2 SIMULINK 25

3.2.1 Simulink diagram of Cognitive Radio 25

3.2.2 PU block diagram 26

3.2.3 Spectrum Allocation using Simulink 27

**CHAPTER-4 NON-CSS TECHNIQUE (Single Threshold)**

**USING MATLAB (28-38)**

4.1 Energy Detection 28

4.1.1 Binary Hypothesis Testing Problem 28

4.1.2 Test Statistic 29

4.1.3 Signal Models 31

4.1.4 Distribution of Test Statistics 32

4.2 Design Parameters 32

4.2.1 Threshold 33

4.2.2 Number of Samples 33

4.3 Simulation Results 33

4.3.1 The plot of Probability of false alarm versus

Probability of detection 34

4.3.2 The plot of Probability of detection versus

Signal to Noise Ratio 35

4.3.3 The plot of Threshold Versus Probability of false alarm 37

**CHAPTER-5 CSS TECHNIQUE (Double threshold) USING MATLAB (39-46)**

5.1 Cooperative Spectrum Sensing (CSS) 39

5.2 Proposed Double Threshold Energy Detection in CSS 39

5.3 Simulation Results 42

**CHAPTER-6 CONCLUSION (46)**

6.1 Spectrum Allocation 46

6.2 Spectrum Sensing 46

**CHAPTER-7 REFERENCES 47**

**APPENDIX (48-74)**

MATLAB CODES 48

1 Spectrum allocation MATLAB code 48

2 Non-Cooperative Spectrum Sensing (Single Threshold)

MATLAB code 55

2.1 Plot of Probability of false alarm vs Probability

of detection 55

2.2 Plot of SNR vs Probability of detection 56

2.3 Plot of threshold vs Probability of false alarm 58

3 Cooperative Spectrum Sensing (Double Threshold)

MATLAB code 59

3.1 Probability of detection vs Probability of detection

for a single energy detector code 59

3.2 Probability of detection vs Probability of false alarm

for a single energy detector code 62

3.3 Cooperative probability of false alarm vs cooperative

probability of detection code 65

3.4 Pe vs SNR & Qd vs SNR code 69

**LIST OF FIGURES**

**Figure No** **Figure** **Name** **Page no**

1.1 Radio communication system before &

after cognitive radio 1

1.2 Ideal SDR system vs Non-Ideal SDR Receiver 2

1.3 Scientific block diagram Of Cognitive Radio 3

1.4 Block diagram of Cognitive Radio Transmitter 3

1.5 Block diagram of Cognitive Radio Receiver 4

1.6 Basic Cognitive cycle 5

1.7 Types of Spectrum sensing techniques 6

1.8 Flowchart of Monte Carlo simulation 9

3.1 Spectrum measurement across the 900KHz-1GHz

band 21

3.2 Block diagram of simulation test-bed 22

3.3 Addition of primary user in the frequency spectrum

in command window 22

3.4 Power spectral density curve 22

3.5 Assigning new user to the Spectrum Holes in

command window 23

3.6 Power spectral density curve: one slot remaining

in the frequency spectrum 23

3.7 Power spectral density curve: All the frequency

bands are efficiently in use 24

3.8 Power spectral density curve: From the frequency

spectrum 3rd slot has been eliminated 24

3.9 Block diagram of cognitive radio taking 5 users

at a time 25

3.10 Block diagram of Primary User (PU) 26

3.11 Spectrum allocation of a primary Users (PU’s) 27

4.1 Energy detector block diagram 28

4.2 Threshold setting in ED: trade-off between missed

detection and false alarm 29

4.3 ROC curve for Pf vs Pd Energy detector-based

spectrum sensing at SNR=-12dB 34

4.4 Plot between Pf and Pd at SNR=-10dB 34

4.5 Plot for Pd vs SNR Energy detector-based spectrum

Spectrum sensing at N=1000 35

4.6 Plot for Pd vs SNR at N=3000 36

4.7 Plot for Pd vs SNR at N=5000 36

4.8 Plot for Pd vs SNR at N=7000 37

4.9 plot for Threshold vs Probability of false alarm 38

5.1 Cooperative spectrum sensing 39

5.2 a. Conventional energy detector with a

single threshold 40

b. Double threshold-based energy detector 40

5.3 Flow chart of the proposed scheme 41

5.4 Pd vs Pfa for a single energy detector 42

5.5 Pd vs Pfa for a single energy detector 43

5.6 Pd vs Pfa for conventional CSS and proposed CSS 43

5.7 Qd vs SNR (in dB) 44

5.8 Pe vs SNR (in dB) 44

**LIST OF TABLES**

**Table No Table page no**

4.1 Improvement in Probability of detection with

increase in SNR in Energy detection method

for AWGN channel 35

4.2 Improvement in Probability of detection with

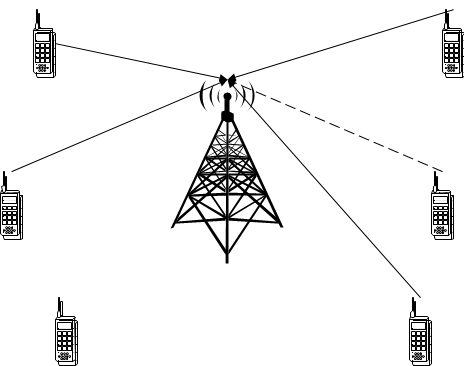
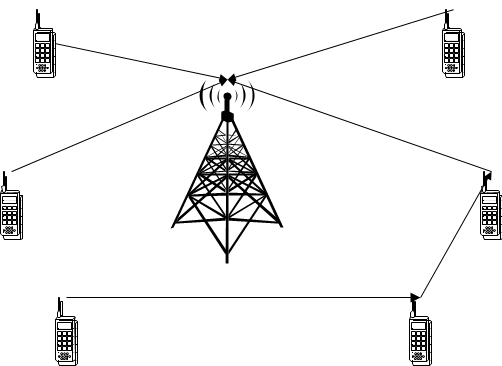
increase in number of samples in Energy detection

method for AWGN channel 37

## CHAPTER 1-INTRODUCTION

In the prosperous world we are living right now, communications enter our daily lives in manifold ways that it is easy to overlook the multitude of its facets. Mobile phones, radio broadcasting, TV towers, satellite antennas, and PCs with the access of the Internet seem to rule the communications world. Data communication networks are a crucial system to any modern city that is because they are used extensively in numerous applications such as financial transactions, social interaction, education, national security, and other more. Although passive optical networks (PONs) may be the best solution for a complex network that requires a high-speed data communication, but the design of the system suffers from high costs and other fiber problems, thus wireless communications solve these troubles. However, there are some problems in wireless communications such as, frequency dependent, relatively low bandwidth, and tightly licensed by the government.

Mobile devices are only allowed to certain frequencies which are getting crowded. With cognitive radio technology, we can use all available frequencies even though those dedicated to TV or satellites. The intelligent devices negotiate in order to use the whole radio spectrum in the most efficient way, this way we can multiply the current network needs. By the word radio we mean any kind of wireless communications, at the moment radios can communicate only with other radios of the same kinds. Cognitive radio can understand the language of any radio, this combined with new single radios embedded in any object, would allow any interaction any physical objects, this can also provide solutions for communications between people at different languages and cultures. In any big events with tens of thousands of people,



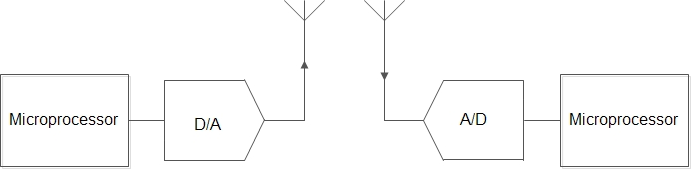
(a) Before Cognitive Radio (b) After Cognitive Radio

Fig. 1.1 Radio communications systems before and after cognitive radio

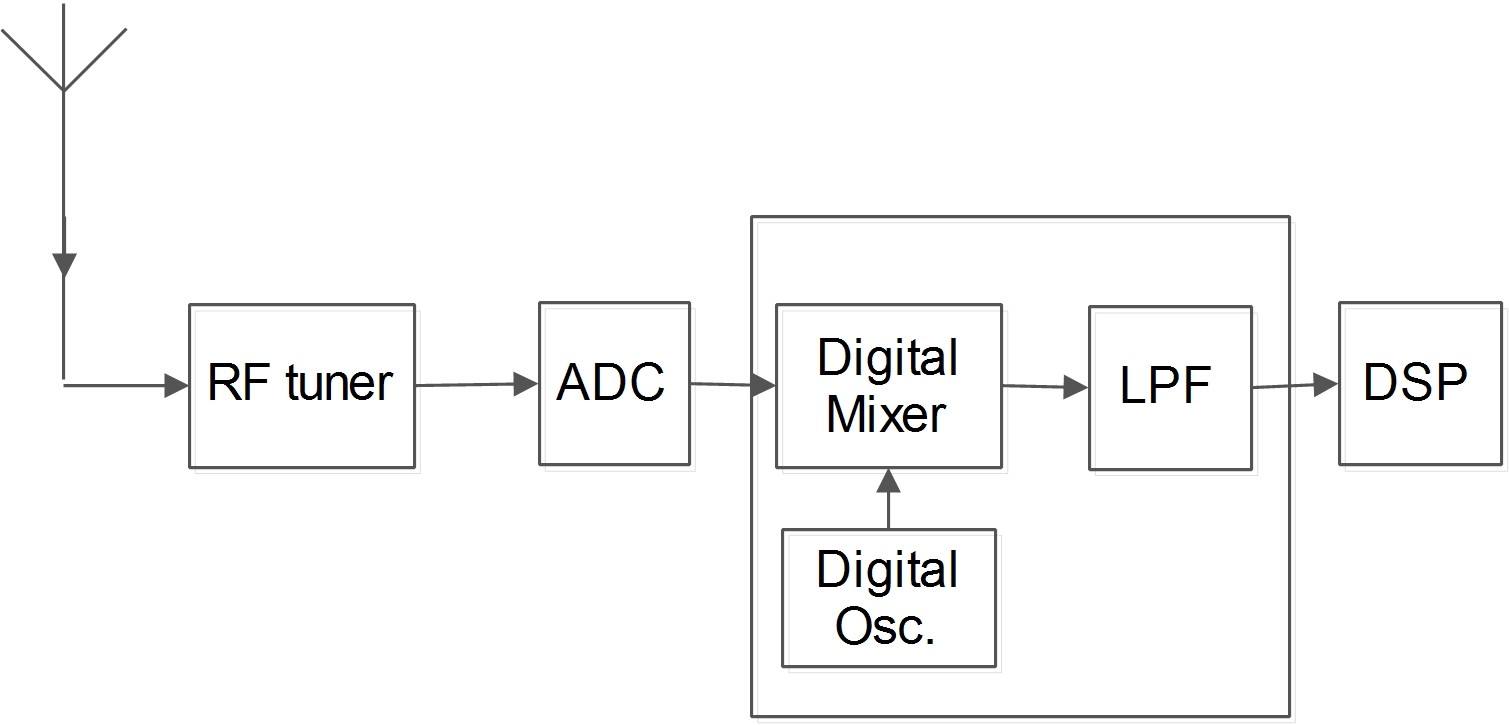
the local network may get overloaded, with current spectrum usage limitations the provided capacity just is not enough for the users. Cognitive radio can use all available frequencies and connectivity methods, it can quickly adapt to unusual situation and ensure the proper operations of the networks. The devices connect not only to through the network cells but also by forming spontaneous networks, this way real time video streams can be transmitted by many more users. Fig.1.1 illustrates the difference between radio communications systems before and after cognitive radio. Before cognitive radio, radio devices sometimes cannot reach the radio tower, it can be seen as dotted line, but after cognitive radio, radio devices are able to connect with each other until it reaches the radio tower.

* 1. **Software Defined Radio (SDR):**

A Software Defined Radio is a radio in which some or all of the physical layer functions are software defined. The ideal (SDR) is shown in Fig. 1.2.a The user data is mapped to the desired waveform in the microprocessor. The digital samples are then converted directly into an RF signal and sent to the antenna. The transmitted signal enters the receiver at the antenna, is sampled and digitized, and finally processed in real time by a general-purpose processor. Generally, its physical components were only an antenna and an Analog Digital Converter (ADC) on the receiver side. Likewise, the transmitter would have a Digital Analog Converter (DAC) and at transmitting antenna Fig. 1.2.b shows non-ideal (SDR) Receiver. At first, the RF tuner converts the analog signal to IF, next, the IF signal is passed to the ADC converter in charge of changing the signal’s domain, offering digital samples. The samples are feed to the following stage’s input which is a Digital Down Converter (DDC).



(a)Ideal SDR



* + 1. Non-Ideal SDR Receiver

Fig. 1.2 Ideal SDR system versus Non-Ideal SDR Receiver

**1.2 Cognitive Radio (CR):**

A cognitive radio is the key technology that allows a cognitive wireless terminal to dynamic access the available spectral opportunities. For cognitive radio, an SDR can take advantage of underutilized spectrum. The definition of cognitive radio by Mitola was generalized by the Federal Communications Commission (FCC) to be “a radio or system that sense its electromagnetic environment and can dynamically and autonomously adjust its radio operating parameters to modify system operation, such as maximize throughput, mitigate interference, facilitate interoperability, access secondary markets".

Despite the differences in the last definitions, a cognitive radio has two key features that distinguish it from a traditional radio: *the cognition capability* (intelligent adaptive behavior) and *the reconfigurability.*

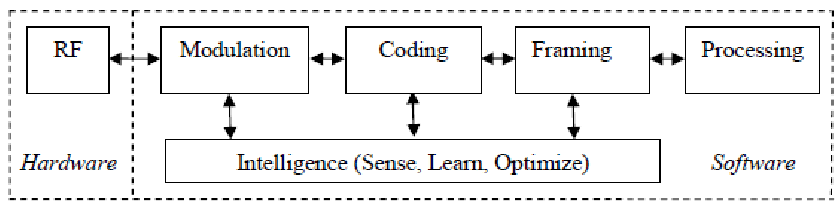
**

Fig. 1.3 Scientific block diagram of Cognitive Radio

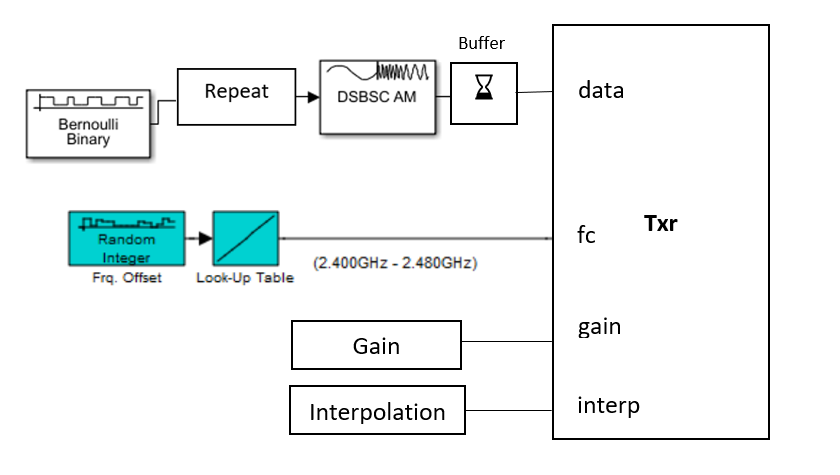
**

Fig. 1.4 Block diagram of Cognitive Radio Transmitter

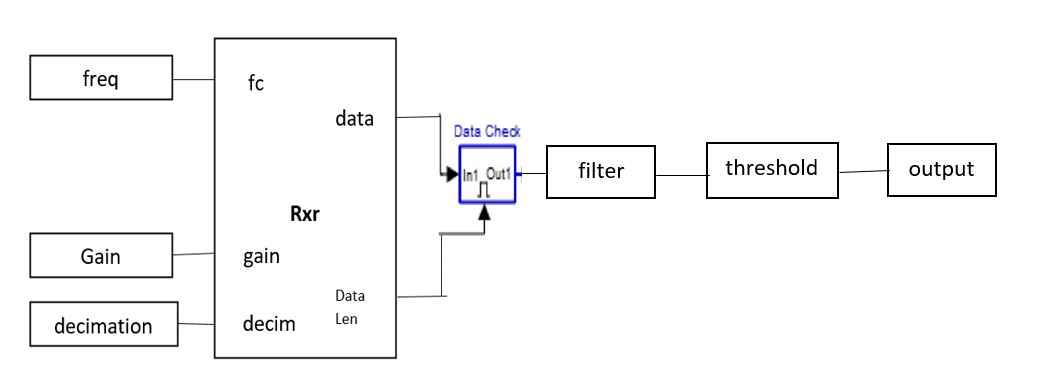


Fig. 1.5 Block diagram of Cognitive Radio Receiver

**1.Bernoulli Binary Generator:** The Bernoulli Binary Generator block generates random binary numbers using a Bernoulli distribution. Use this block to generate random data bits to simulate digital communication systems and obtain performance metrics such as bit error rate. The Bernoulli distribution with parameter p produces zero with probability p and one with probability 1-p. The Bernoulli distribution has mean value 1-p and variance p(1-p). The **Probability of zero** parameter specifies p and can be any real number in range [0, 1].

**2. DSBSC AM Mod:** The DSB AM Modulator Passband block modulates using double-sideband amplitude modulation. The output is a passband representation of the modulated signal. Both the input and output signals are real scalar signals.

If the input is u(t) as a function of time t, then the output is

(*u*(*t*)+*k*)cos(2*πfct*+*θ*)

where:

* k is the **Input signal offset** parameter.
* fc is the **Carrier frequency** parameter.
* θ is the **Initial phase** parameter

**3.Random Integer Generator:** The Random Integer Generator block generates uniformly distributed random integers in the range [0, M-1], where M is specified by the [Set size](https://in.mathworks.com/help/comm/ref/randomintegergenerator.html#fp102455_Set_size) parameter. Use this block to generate random binary-valued or integer-valued data.

**4. Look-up Table:** The 1-D, 2-D, and n-D Lookup Table blocks evaluate a sampled representation of a function in N variables

*y*=*F* (*x*1, *x*2, *x*3,,,,*xN*)

where the function F can be empirical. The block maps inputs to an output value by looking up or interpolating a table of values you define with block parameters. The block supports flat (constant), linear (linear point-slope), Lagrange (linear Lagrange), nearest, cubic-spline, and Akima spline interpolation methods. You can apply these methods to a table of any dimension from 1 through 30.

**5.Threshold Comparator:** A comparator is an electronic circuit, which compares the two inputs that are applied to it and produces an output. The output value of the comparator indicates which of the inputs is greater or lesser. Please note that comparator falls under non-linear applications of ICs.

Example of cognitive radio transmitter is: USRP2 transmitter and receiver which has variable gain and variable interpolation. For receiver, non-ideal software defined radio receiver is used which is shown in fig:1.2. Non-ideal SDR is a super heterodyne receiver.

**1.3 Functions of Cognitive Radio:**

There are four major functions of Cognitive Radio. Figure 1.6. shows the basic cognitive cycle

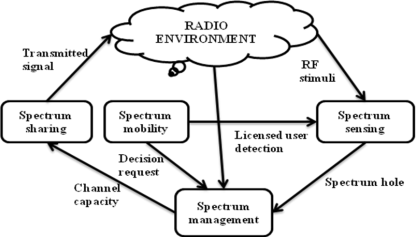


Figure 1.6. Basic cognitive cycle

**1.3.1 Spectrum Sensing**

The first step of spectrum sensing is that it determines the presence of primary user on a band. The cognitive radio is able to share the result of its detection with other cognitive radios after sensing the spectrum. The goal of spectrum sensing is to find out the spectrum status and activity by periodically sensing the target frequency band. Particularly, a cognitive radio transceiver detects the spectrum which is unused or spectrum hole and also determines method of access without interfering the transmission of licensed. Two types of spectrum sensing techniques are there; those are 1. Signal processing techniques (or) non-Cooperative Spectrum Sensing 2. Cooperative Spectrum Sensing as shown below fig 1.7.

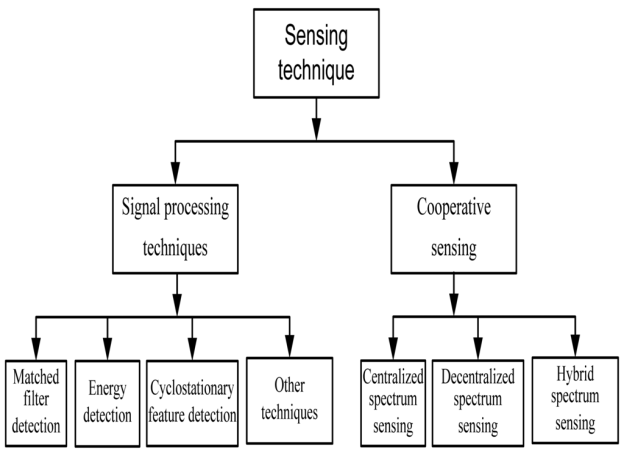


Fig:1.7 Types of Spectrum Sensing Techniques

**1.3.1.1 Non-Cooperative Spectrum Sensing Techniques:**

In a realistic spectrum sensing scenario, there are situations in which only one sensing terminal is available or in which no cooperation is allowed due to the lack of communication between sensing terminals. In this section we will explore the main single user sensing schemes, some of which will serve as basis for the development of the cooperative ones. Single user spectrum sensing approaches have been heavily studied in the literature, in part because of the relationship to signal detection. There are several classical techniques for this purpose, Mainly the energy detector (ED), Matched-filter detection and Cyclostationary detection.

**1.3.1.2** **Cooperative Spectrum Sensing Techniques:**

The cooperative spectrum sensing can be:

* Centralized, in which a central entity gathers all information from all secondary receivers to make a decision about the medium status, which is then transmitted back to the receivers
* Distributed, in which the receivers share their information in order to make their own decision

### **Spectrum management**

Spectrum Management: Provides the fair spectrum scheduling method among coexisting users. The available white space or channel is immediately selected by cognitive radio if once found. This property of cognitive radio is described as spectrum management. Spectrum sensing, spectrum analysis, and spectrum decision fall in spectrum Management. Spectrum Sensing has been discussed in previous section. Spectrum Analysis makes possible the characterization of different spectrum bands, which is exploited to get the spectrum band appropriate requirements of the user. Spectrum decision refers to a cognitive radio decides the data rate, determines the transmission mode, and the transmission bandwidth. Then, the appropriate spectrum band is selected according to the spectrum characteristics and user requirements.

### **Spectrum Sharing**

Cognitive Radio assigns the unused spectrum (spectrum hole) to the secondary user (SU) as long as primary user (PU) does not use it. This property of cognitive radio is described as spectrum sharing.

**Underlay Spectrum Sharing**: Underlay spectrum sharing is the availability of the radio spectrum access with minimal transmission power that the interference temperature above its pre-designed thresholds wouldn’t be raised. To spread the unlicensed signal over a large band of spectrum in underlay spectrum sharing the licensed radio device can identify undesired signal which is below the noise and interference floor. Overlay Spectrum sharing: Unlicensed users can utilize a spectrum band for the fraction of time where this band is under-utilized by the licensed users in Overlay Spectrum sharing technique.

### **Spectrum Mobility**

When a licensed (Primary) user is detected the Cognitive Radio (CR) vacates the channel. This property of cognitive radio is described as the spectrum mobility and also called handoff. This is the process that allows the Cognitive Radio user to change its operating frequency. Cognitive Radio networks try to use the spectrum dynamically to operate in the best available frequency band and maintain the transparent communication. Spectrum sensing is an important and a sensitive job out of these four functions in Cognitive Radio since interfering with other users is illegal.

### **1.4 Types of Cognitive Radio**

There are two types of Cognitive Radios:

**Full Cognitive Radio**: Full Cognitive Radio (CR) considers all parameters. A wireless node or network can be conscious of every possible parameter observable.

**Spectrum Sensing Cognitive Radio**: Detects channels in the radio frequency spectrum. Fundamental requirement in cognitive radio network is spectrum sensing. To enhance the detection probability many signal detection techniques are used in spectrum sensing.

The performances for cognitive radio system require: i) authentic spectrum hole and detection of primary user, ii) precise link estimation between nodes, iii) fast and accurate frequency control and iv) method of power control that assures reliable communication between cognitive radio terminals and non-interference to the primary users.

### **1.5 Characteristics of Cognitive Radio:**

There are two main characteristics of the cognitive radio and can be defined

* **Cognitive capability:** Cognitive Capability defines the ability to capture or sense the information from its radio environment of the radio technology. Joseph Mitola first explained the cognitive capability in term of the cognitive cycle “a cognitive radio continually observes the environment, orients itself, creates plans, decides, and then acts”
* **Reconfigurability**: Cognitive capability offers the spectrum awareness, Reconfigurability refers to radio capability to change the functions, enables the cognitive radio to be programmed dynamically in accordance with radio environment (frequency, transmission power, modulation scheme, communication protocol.

### **1.6 Monte Carlo Simulation Technique:**

Monte Carlo simulation is a computerized mathematical technique to generate random sample data based on some known distribution for numerical experiments. This method is applied to risk quantitative analysis and decision-making problems. This method is used by the professionals of various profiles such as finance, project management, energy, manufacturing, engineering, research & development, insurance, oil & gas, transportation, etc.

Monte Carlo Simulation ─ Important Characteristics

* Its output must generate random samples.
* Its input distribution must be known.
* Its result must be known while performing an experiment.

Monte Carlo Simulation ─ Advantages

* Easy to implement.
* Provides statistical sampling for numerical experiments using the computer.
* Provides approximate solution to mathematical problems.

The following illustration shows a generalized flowchart of Monte Carlo simulation.

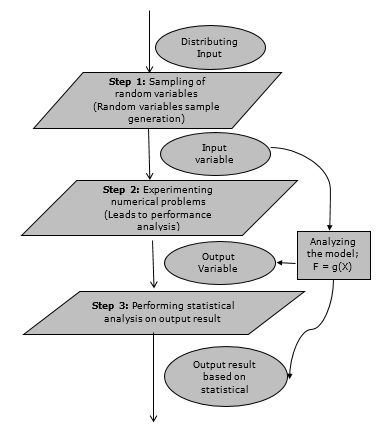
****

Fig.1.8 flowchart of Monte Carlo simulation

**CHAPTER 2-SOFTWARE DESCRIPTION**

**2.1 Introduction to MATLAB**

MATLAB is an elite dialect for specialized registering. It incorporates calculation, representation, and programming in a simple to-utilize condition where issues and arrangements are communicated in natural numerical documentation. Run of the mill utilizes incorporate

• Math and calculation

• Algorithm advancement

• Data obtaining

• Modeling, reenactment, and prototyping

• Data examination, investigation, and representation

• Scientific and designing illustrations

• Application advancement, including graphical UI building

MATLAB is an intuitive framework whose important records thing is an exhibit that doesn't require dimensioning. This allows you to address several specialized processing issues, specifically people with framework and vector data, in a small quantity of the time it might take to compose a program in a scalar non intuitive dialect, as an instance, C or FORTRAN.

The name MATLAB stays for grid studies facility. MATLAB modified into to start with composed to offer simple access to framework programming created thru the LINPACK and EISPACK ventures. Today, MATLAB vehicles fuse the LAPACK and BLAS libraries, putting the cutting element in programming for network calculation.

MATLAB has advanced over a time of years with contribution from several customers. In college situations, it's miles the same old educational system for early on and propelled courses in arithmetic, designing, and technological know-how. In industry, MATLAB is the tool of selection for high-profitability research, development, and exam.

MATLAB highlights a collection of extra utility-precise preparations called device booths. Important to most clients of MATLAB, tool kits allow you to examine and practice particular innovation. Tool compartments are exhaustive accumulations of MATLAB capacities (M-records) that extend out the MATLAB situation to take care of precise instructions of problems. Territories in which device stash is reachable incorporate flag dealing with, manage frameworks, neural systems, fluffy purpose, wavelets, recreation, and several others.

The tutorials are independent of the rest of the document. The primarily objective is to help you learn quickly the first steps. The emphasis here is “learning by doing”. Therefore, the best way to learn is by trying it yourself. Working through the examples will give you a feel for the way that MATLAB operates. In this introduction we will describe how MATLAB handles simple numerical expressions and mathematical formulas. The name MATLAB stands for matrix laboratory. MATLAB was written originally to provide easy access to matrix software developed by the LINPACK (linear system package) and EISPACK (Eigen system package) projects. MATLAB [1] is a high-performance language for technical computing. It integrates computation, visualization, and programming environment. Furthermore, MATLAB is a modern programming language environment: it has sophisticated data structures, contains built-in editing and debugging tools, and supports object-oriented programming. These factors make MATLAB an excellent tool for teaching and research. MATLAB has many advantages compared to conventional computer languages (e.g., C, FORTRAN) for solving technical problems. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. The software package has been commercially available since 1984 and is now considered as a standard tool at most universities and industries worldwide. It has powerful built-in routines that enable a very wide variety of computations. It also has easy to use graphics commands that make the visualization of results immediately available. Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering.

**2.2 Basic building blocks of MATLAB**

The basic building block of MATLAB is MATRIX. The fundamental data type is the array. Vectors, scalars, real matrices and complex matrix are handled as specific class of this basic data type. The built-in functions are optimized for vector operations. No dimension statements are required for vectors or arrays.

**2.2.1 MATLAB Window**

The MATLAB works based on five windows: Command window, Workspace window, Current directory window, Command history window, Editor Window, Graphics window and Online-help window.

**2.2.2 Command Window**

The command window is where the user types MATLAB commands and expressions at the prompt (>>) and where the output of those commands is displayed. It is opened when the application program is launched. All commands including user-written programs are typed in this window at MATLAB prompt for execution.

**2.2.3 Work Space Window**

MATLAB defines the workspace as the set of variables that the user creates in a work session. The workspace browser shows these variables and some information about them. Double clicking on a variable in the workspace browser launches the Array Editor, which can be used to obtain information.

**2.2.4 Current Directory Window**

The current Directory tab shows the contents of the current directory, whose path is shown in the current directory window. For example, in the windows operating system the path might be as follows: C:\MATLAB\Work, indicating that directory “work” is a subdirectory of the main directory “MATLAB”; which is installed in drive C. Clicking on the arrow in the current directory window shows a list of recently used paths. MATLAB uses a search path to find M-files and other MATLAB related files. Any file run in MATLAB must reside in the current directory or in a directory that is on search path.

**2.2.5 Command History Window**

The Command History Window contains a record of the commands a user has entered in the command window, including both current and previous MATLAB sessions. Previously entered MATLAB commands can be selected and re-executed from the command history window by right clicking on a command or sequence of commands.

This is useful to select various options in addition to executing the commands and is useful feature when experimenting with various commands in a work session.

**2.2.6 Editor Window**

The MATLAB editor is both a text editor specialized for creating M-files and a graphical MATLAB debugger. The editor can appear in a window by itself, or it can be a sub window in the desktop. In this window one can write, edit, create and save programs in files called M-files.

MATLAB editor window has numerous pull-down menus for tasks such as saving, viewing, and debugging files. Because it performs some simple checks and also uses color to differentiate between various elements of code, this text editor is recommended as the tool of choice for writing and editing M-functions.

**2.2.7 Graphics or Figure Window**

The output of all graphic commands typed in the command window is seen in this window.

**2.2.8 Online Help Window**

MATLAB provides online help for all it’s built-in functions and programming language constructs. The principal way to get help online is to use the MATLAB help browser, opened as a separate window either by clicking on the question mark symbol (?) on the desktop toolbar, or by typing help browser at the prompt in the command window. The help Browser is a web browser integrated into the MATLAB desktop that displays a Hypertext Markup Language (HTML) documents. The Help Browser consists of two panes, the help navigator pane, used to find information, and the display pane, used to view the information. Self-explanatory tabs other than navigator pane are used to perform a search.

**2.3 The MATLAB System**

The MATLAB machine includes five predominant parts.

**2.3.1 Development Environment**

This is the set of tools and facilities that assist you use MATLAB functions and files. Many of those systems are graphical consumer interfaces. It includes the MATLAB computing device and Command Window, a command history, an editor and debugger, and browsers for viewing help, the workspace, files, and the hunt direction.

**2.3.2 The MATLAB Mathematical Function**

This is a wonderful collection of computational algorithms starting from essential features like sum, sine, cosine, and complex mathematics, to extra sophisticated features like matrix inverse, matrix Eigen values, Bessel competencies, and fast Fourier transforms.

**2.3.3 The MATLAB Language**

This is an excessive-degree matrix/array language with manipulate go with the go with the flow statements, competencies, statistics systems, input/output, and object-oriented programming talents. It permits every "programming in the small" to hastily create quick and dirty throw-away programs, and "programming in the big" to create entire massive and complicated software applications.

**2.3.4 Graphics**

MATLAB has huge facilities for showing vectors and matrices as graphs, further to annotating and printing those graphs. It consists of immoderate-degree capabilities for two-dimensional and three-dimensional facts visualization, image processing, animation, and presentation images. It additionally includes low-stage functions that allow you to completely customize the appearance of image graphs further to assemble whole graphical consumer interfaces on your MATLAB packages.

**2.3.5 The MATLAB Application Program Interface (API)**

This is a library that lets in you to write C and Fortran packages that engage with MATLAB. It includes centers for calling workout routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for analyzing and writing MAT-documents.

**2.4 MATLAB Files**

MATLAB has three types of files for storing information. They are: M-files and MAT-files.

**2.4.1 M-Files**

These are standard ASCII text file with ‘m’ extension to the file name and creating own matrices using M-files, which are text files containing MATLAB code. MATLAB editor or another text editor is used to create a file containing the same statements which are typed at the MATLAB command line and save the file under a name that ends in .m.

There are two types of M-files:

**2.4.1.1 Script Files**

It is an M-file with a set of MATLAB commands in it and is executed by typing name of file on the command line. These files work on global variables currently present in that environment.

**2.4.1.2 Function Files**

A function file is also an M-file except that the variables in a function file are all local. This type of files begins with a function definition line.

**2.5 MAT-Files**

These are binary data files with. mat extension to the file that are created by MATLAB when the data is saved. The data written in a special format that only MATLAB can read. These are located into MATLAB with ‘load’ command.

**2.6 MATLAB Working Environments**

**2.6.1 MATLAB Desktop**

MATLAB Desktop is the precept MATLAB utility window. The computing device consists of five sub windows, the summon window, the workspace software, the prevailing catalog window, the order records window, and at the least one figure home windows, which are confirmed simply whilst the patron indicates a realistic.

The order window is the vicinity the consumer types MATLAB orders and expressions on the provoke (>>) and where the yield of these costs is shown. MATLAB characterizes the workspace because the association of factors that the purchaser makes in a work consultation. The workspace application demonstrates these elements and a few information approximately them. Double tapping on a variable in the workspace software dispatches the Array Editor, which can be applied to get statistics and wage instances regulate certain homes of the variable.

The gift Directory tab over the workspace tab demonstrates the substance of the prevailing registry, whose way is seemed within the gift index window. 1For case, in the home windows working framework the manner may be as consistent with the subsequent: C: MATLAB Work, demonstrating that registry "work" is a subdirectory of the number one catalog "MATLAB", that's added in force C. Tapping on the bolt inside the gift index window demonstrates a rundown of as of late utilized approaches. Tapping at the capture to at least one aspect of the window allows the customer to exchange the prevailing catalog.

MATLAB utilizes an inquiry way to discover M-information and other MATLAB associated documents, which might be kind out in catalogs within the PC report framework. Any report hold going for walks in MATLAB must reside in the ebb and go with the flow registry or in an index this is on are looking for manner. Of course, the facts provided with MATLAB and math works device kits are integrated into the inquiry manner. The least traumatic approach to look which indexes are at the inquiry manner. The best method to peer which catalogs are quickly the search way, or to consist of or adjust an inquiry manner, is to pick out set way from the File menu the computing device, and after that utilization the set way change container. It is extremely good exercise to feature any normally applied catalogs to the pursuit manner to preserve a strategic distance from again and again having the exchange the existing index.

The Command History Window incorporates a file of the orders a customer has entered in the rate window, together with each gift and beyond MATLAB classes. Already entered MATLAB orders may be selected and re-done from the fee history window by using right tapping on a summon or association of orders. This interest dispatches a menu from which to choose exclusive picks however executing the orders. This is helpful to pick special choices notwithstanding executing the summons. This is a precious element even as attempting different things with distinct orders in a work consultation.

**2.6.2 Using the MATLAB Editor to create M-Files**

The MATLAB manager is both a word processor specific for making M-information and a graphical MATLAB debugger. The proofreader can display up in a window without everybody else, or it could be a sub window inside the desktop. M-statistics are supposed by means of the growth .M, as in pixel up. m. The MATLAB editorial manager window has diverse draw down menus for errands, as an example, sparing, seeing, and troubleshooting files. Since it plays out some primary tests and furthermore makes use of shading to separate among exceptional components of code, this content material device is suggested because the apparatus of choice for composing and changing M-capacities. To open the proofreader, sort adjust at the incite opens the M-report filename. m in a supervisor window, organized for changing. As referred to before, the file have to be within the momentum catalog, or in an index inside the pursuit way.

**2.7 Basic Commands**

Pwd prints working directory

Demo demonstrates what is possible in **Mat lab**

Who lists all of the variables in your Mat lab workspace?

Whose list the variables and describes their matrix size

Clear erases variables and functions from memory

Clear x erases the matrix 'x' from your workspace

Close by it, closes the current figure window

Figure creates an empty figure window

Hold on holds the current plot and all axis properties so that subsequent graphing

Commands add to the existing graph

Hold off sets the next plot property of the current axes to "replace"

Find indices of nonzero elements

e.g.: d = find(x>100) returns the indices of the vector x that are greater than 100

Break terminate execution of m-file or WHILE or FOR loop

For repeat statements a specific number of times, the general form of a FOR

Statement is:

FOR variable = expr, statement, ..., statement END

For n=1:cc/c;

Magn (n,1)=NaNmean(a((n-1)\*c+1:n\*c,1));

End

Diff difference and approximate derivative e.g.:

DIFF(X) for a vector X, is [X(2)-X(1) X(3)-X(2) ... X(n)-X(n-1)].

Nan the arithmetic representation for Not-a-Number, a NaN is obtained as a

Result of mathematically undefined operations like 0.0/0.0

INF the arithmetic representation for positive infinity, a infinity is also produced by operations like dividing by zero, e.g. 1.0/0.0, or from overflow, e.g. exp(1000).

Save saves all the matrices defined in the current session into the file,

matlab.mat, located in the current working directory

Load loads contents of matlab.mat into current workspace

Save filename x y z saves the matrices x, y and z into the file titled filename.mat

Save filename x y z /ASCII save the matrices x, y and z into the file titled filename.dat

Load filename loads the contents of filename into current workspace; the file can

be a binary (.mat) file

Load filename.dat loads the contents of filename.dat into the variable filename

Xlabel (‘ ’) : Allows you to label x-axis

Ylabel(‘ ‘) : Allows you to label y-axis

Title (‘ ‘) : Allows you to give title for Plot

Subplot (): Allows you to create multiple Plots in the same window

**2.8 Some Basic Plot Commands**

**2.8.1 Kinds of Plots**

Plot(x, y) creates a Cartesian plot of the vectors x & y

Plot(y) creates a plot of y vs. the numerical values of the elements in the y-vector

Semilogx(x,y) plots log(x) vs y

Semi logy(x,y) plots x vs log(y)

loglog(x,y) plots log(x) vs log(y)

Polar (theta) creates a polar plot of the vectors r & theta where theta is in radians

Bar(x) creates a bar graph of the vector x. (Note also the command stairs(x))

Bar(x, y) creates a bar-graph of the elements of the vector y, locating the bars

According to the vector elements of 'x'

**2.8.2 Plot Description**

Grid creates a grid on the graphics plot

Title ('text') places a title at top of graphics plot

Xlabel ('text') writes 'text' beneath the x-axis of a plot

Ylabel ('text') writes 'text' beside the y-axis of a plot

Text(x, y,'text') writes 'text' at the location (x, y)

Text(x, y,'text','sc') writes 'text' at point x,y assuming lower left corner is (0,0)

And upper right corner is (1, 1)

Axis ([xmin xmax ymin ymax]) sets scaling for the x- and y-axes on the current plot

**2.9 Algebraic Operations in Mat lab**

**2.9.1 Scalar Calculations**

+ Addition

- Subtraction

\* Multiplication

/ Right division (a/b means a ÷ b)

\ left division (a\b means b ÷ a)

^ Exponentiation

Array products: Recall that addition and subtraction of matrices involved addition or subtraction of the individual elements of the matrices.

Sometimes it is desired to simply multiply or divide each element of an matrix by the corresponding element of another matrix 'array operations”. Array or element-by-element operations are executed when the operator is preceded by a '.' (Period):

a.\* b multiplies each element of a by the respective element of b

a ./ b divides each element of a by the respective element of b

a .\ b divides each element of b by the respective element of a

a .^ b raise each element of a by the respective b element

**2.10 Introduction to SIMULINK:**

Simulink is a simulation and model-based design environment for dynamic and embedded systems, integrated with MATLAB. Simulink, also developed by MathWorks, is a data flow graphical programming language tool for modelling, simulating and analyzing multi-domain dynamic systems. It is basically a graphical block diagramming tool with customizable set of block libraries.

It allows you to incorporate MATLAB algorithms into models as well as export the simulation results into MATLAB for further analysis.

Simulink supports −

* system-level design
* simulation
* automatic code generation
* testing and verification of embedded systems

There are several other add-on products provided by MathWorks and third-party hardware and software products that are available for use with Simulink.

The following list gives brief description of some of them −

* **State flow** allows developing state machines and flow charts.
* **Simulink Coder** allows the generation of C source code for real-time implementation of systems automatically.
* **xPC Target** together with **x86-based real-time systems** provide an environment to simulate and test Simulink and State flow models in real-time on the physical system.
* **Embedded Coder** supports specific embedded targets.
* **HDL Coder** allows to automatically generate synthesizable VHDL and Verilog.
* **SimEvents** provides a library of graphical building blocks for modelling queuing systems.

Simulink is capable of systematic verification and validation of models through modelling style checking, requirements traceability and model coverage analysis.

Simulink Design Verifier allows you to identify design errors and to generate test case scenarios for model checking.

**2.11 Getting Help**

The important approach to get help online is to utilize the MATLAB help program, opened as a different window either by tapping on the question mark image (?) on the desktop toolbar, or by writing help program at the provoke in the order window. The assistance Browser is a web program coordinated into the MATLAB desktop that shows a Hypertext Markup Language (HTML) records. The Help Browser comprises of two sheets, the assistance pilot sheet, used to discover data, and the show sheet, used to see the data.

## CHAPTER 3- SPECTRUM ALLOCATION BY USING MATLAB and SIMULINK

**3.1 MATLAB**

Matrix Laboratory or [MATLAB for short is](https://www.educba.com/what-is-matlab/) a multi-paradigm numerical computing environment and proprietary programming language developed by MathWorks. It combines computation, visualization, and programming in an easily usable environment and is all expressed in mathematical equations. Written in C, [C++, and Java](https://www.educba.com/c-plus-plus-vs-java/), MATLAB was initially released in 1984. The latest version has been released on March 2018.

Applications of MATLAB is built around the MATLAB scripting language and revolves based on the following mathematical concepts:

* Variables
* Vectors and matrices
* Structures
* Functions
* Function handles
* Classes and object-oriented programming

**3.1.1 Spectrum Holes:** In some locations or at some times of the day, 70 percent of the allocated spectrum may be sitting idle. The Federal Communications Commission of the United States Government (FCC) has recently recommended that significantly greater spectral efficiency could be realized by deploying wireless devices that can coexist with the licensed users.

**3.1.2 Assigning Primary User to the Frequency Spectrum:** We have designed our system to have 5 different frequency channels and each User is assigned a particular frequency band. Once we run our program it will ask to add a User and assign it a particular band in ascending order. Here in [Fig.3.3](#_bookmark157)we have not entered User 2, 3 & 5, thus their respective bands are still un-allocated. We can see them below in the power spectral density graph of our carrier signal.

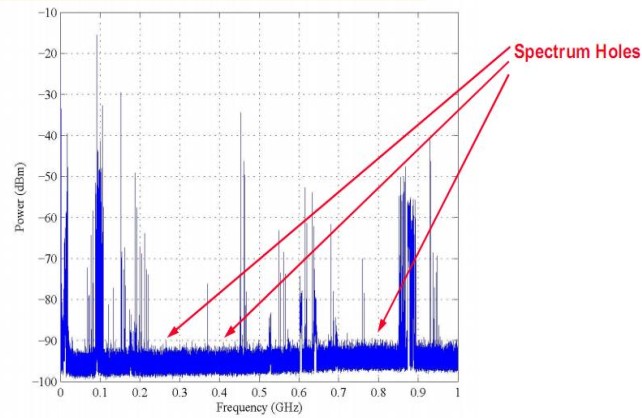


Fig. 3.1 Spectrum measurement across the 900 KHz - 1 GHz band.

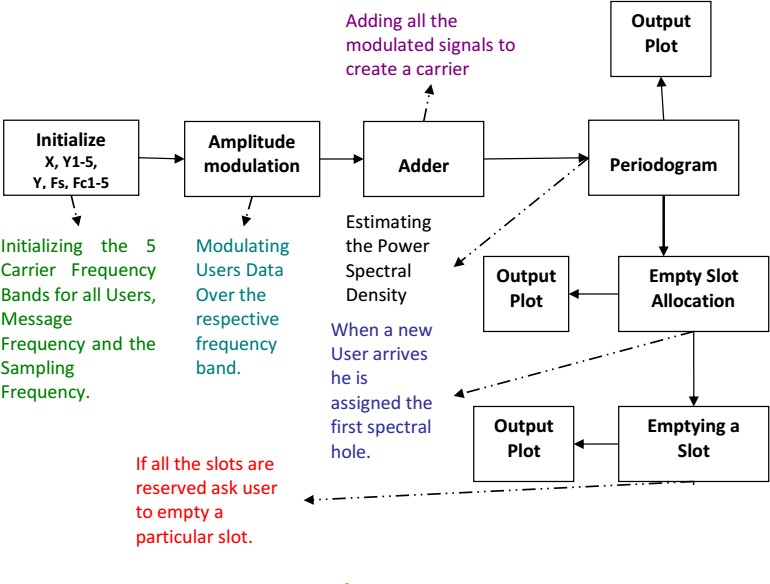


Fig. 3.2 Block diagram of simulation test-bed.

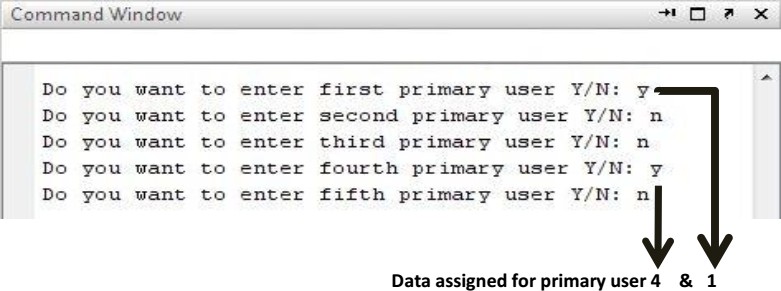


Fig. 3.3 Addition of primary user in the frequency spectrum in Command Window.

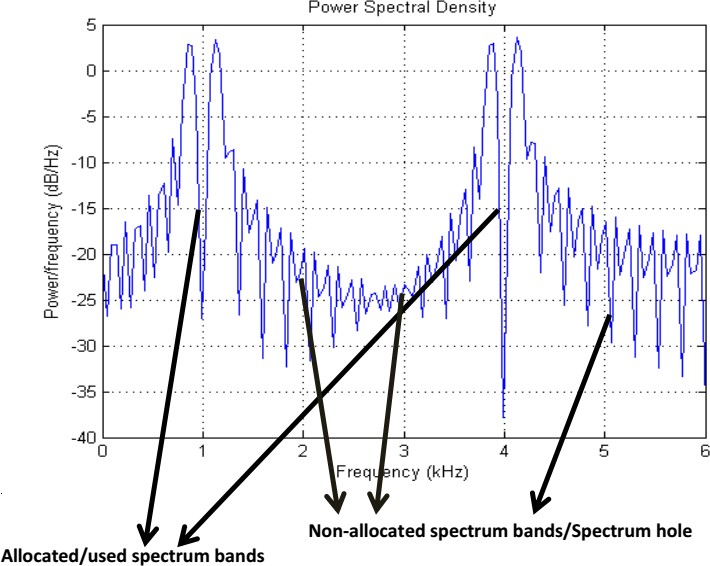


Fig. 3.4 Power spectral density curve.

**3.1.3 Assigning new user to the Spectrum Holes:** Now we are adding another User, the system will search the first available gap in the spectrum and automatically assign it to the new user. As the first available gap was after User-1 as User-2 was not sending any data so the band reserved for User-2 at start is now assigned to this new User.

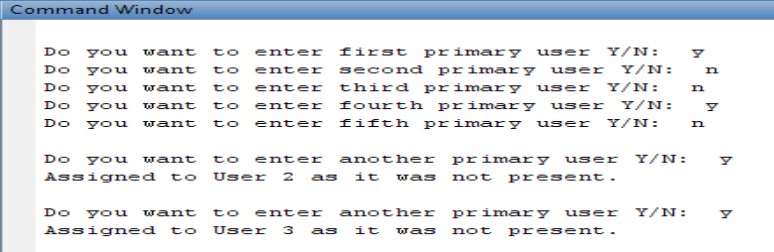


Fig. 3.5 Assigning new user to the Spectrum Holes in Command window.

Here we can see that the first spectral gap has been filled by assigning the new incoming User’s data. The first spectral gap belonged was that of User-2.

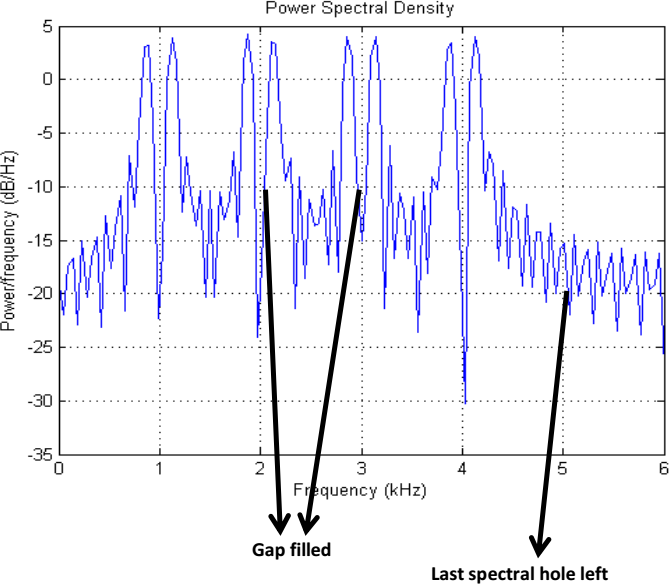


Fig. 3.6 Power spectral density curve: one slot remaining in the frequency spectrum.

**3.1.4 Efficient frequency Band width:** Now we have just one empty slot left which will get filled by addition of another Primary User. The power spectral density curve of the signal shows us that all of the frequency bands are efficiently in use after the addition of the last incoming user.

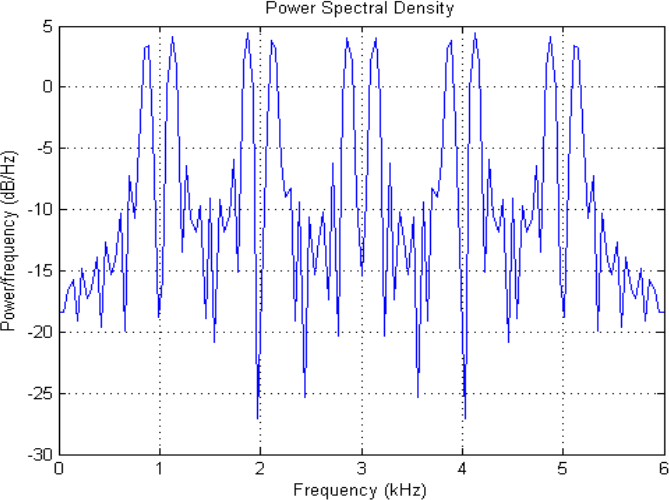


Fig. 3.7 Power spectral density curve: All of the frequency bands are efficiently in use.

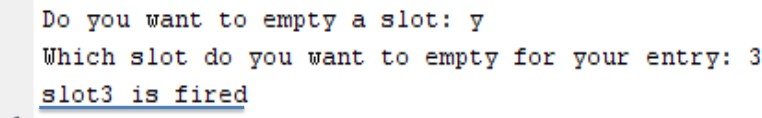
**3.1.5 Elimination of a Slot:** Once all the slots are being assigned, our system will entertain no other User and will be able to free up the slots one by one as shown below. If we ask it to empty a slot, it will remove the data of that slot and make it ready for the next assignment.

Fig. 3.8 Elimination of a slot in Command Window.

Following graph is shown [in fig 3.9.](#_bookmark166)

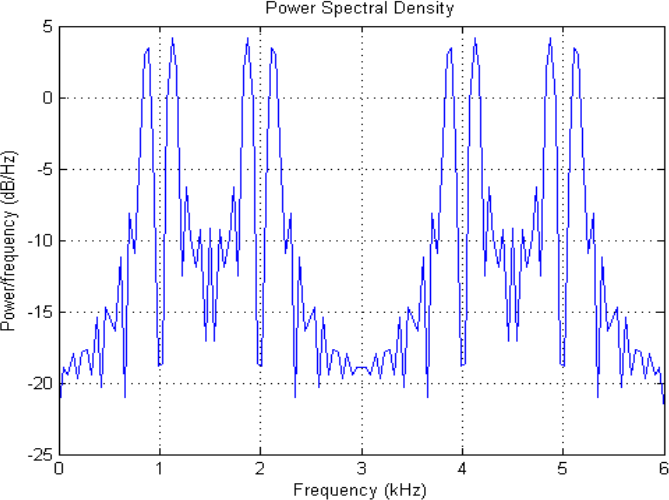


Fig. 3.8 Power spectral density curve: From the frequency spectrum 3rd slot has been eliminated.

**3.2 SIMULINK**

MATLAB Simulink is developed by the MathWorks. The Simulink library browser contains the collection of multiple tools and their functions. It is useful for the simulation of the dynamic system in the MATLAB environment. The Simulink toolboxes provide the specific tools for analyzing, designing, simulation of the system, making the communication between the other system, etc.

**Simulink** **supports** −

* system-level design
* simulation
* automatic code generation
* testing and verification of embedded systems

**3.2.1 Simulink diagram of Cognitive Radio:**

The block diagram of cognitive radio is shown in the fig:3.10. It consists of Primary User (PU), mux, spectrum scope and to workspace blocks.

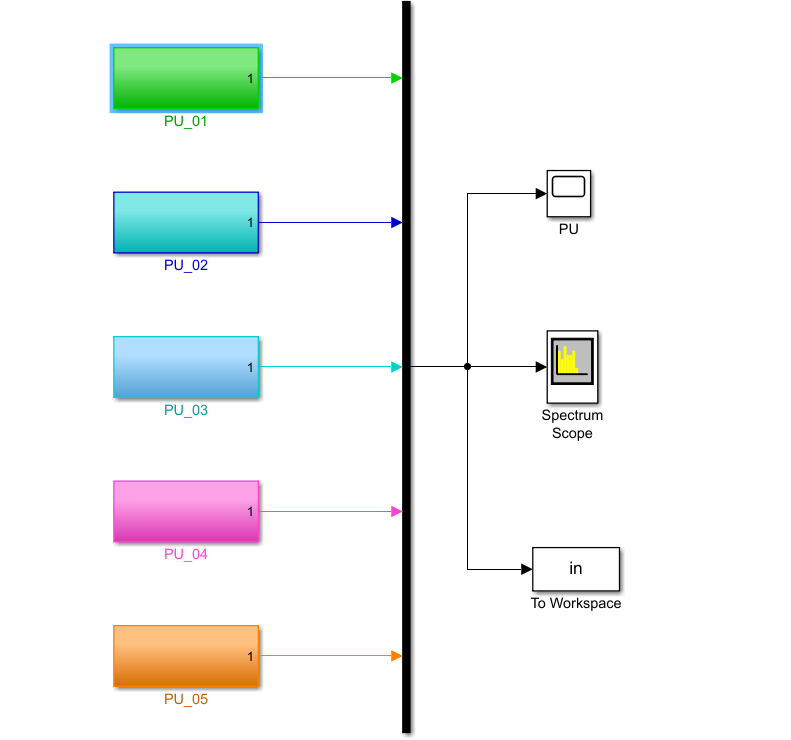
****

Fig. 3.9 Block diagram of cognitive radio taking 5 users at a time.

**1.Mux:** The Mux block combines inputs with the same data type and complexity into a vector output. The output mux signal is flat, even if you create the mux signal from other mux signals. However, you can use multiple Mux blocks to create a mux signal in stages. A mux signal simplifies the visual appearance of a model by combining two or more signal lines into one line. Mux signals do not affect simulation or code generation.

**2. Spectrum Analyzer:** The Spectrum Analyzer block, referred to here as the scope, displays the frequency spectra of signals. You can use the Spectrum Analyzer block in models running in Normal or Accelerator simulation modes. You can also use the Spectrum Analyzer block in models running in Rapid Accelerator or External simulation modes, with some limitations.

**3. To workspace:** The To Workspace block writes input signal data to a workspace. During simulation, the block writes data to an internal buffer. When you pause the simulation or the simulation completes, that data is written to the workspace. Data is not available until the simulation pauses or stops.

**4.Scope:** The two blocks have identical functionality, but different default settings. The Time Scope is optimized for discrete time processing. The Scope is optimized for general time-domain simulation. For a side-by-side comparison.

**5.Primary User:** The PU block describes about the user who is accessing the spectrum at that time. The primary user is as shown in below fig:3.11.

**3.2.2 PU block diagram:**

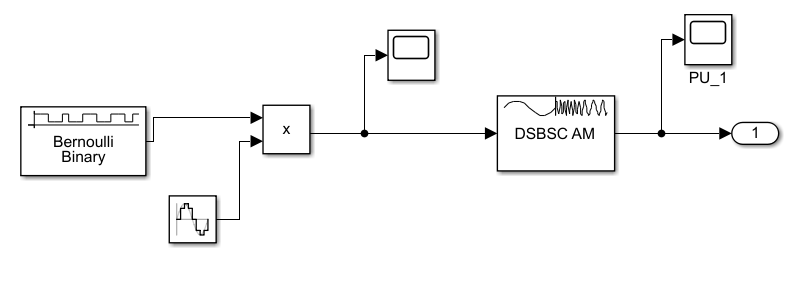
****

Fig. 3.10 Block diagram of Primary User (PU)

**3.2.3 Spectrum Allocation using Simulink:**

The spectrum allocation using the above Simulink model is shown in below fig:3.12 which was taking 5 users at a time.

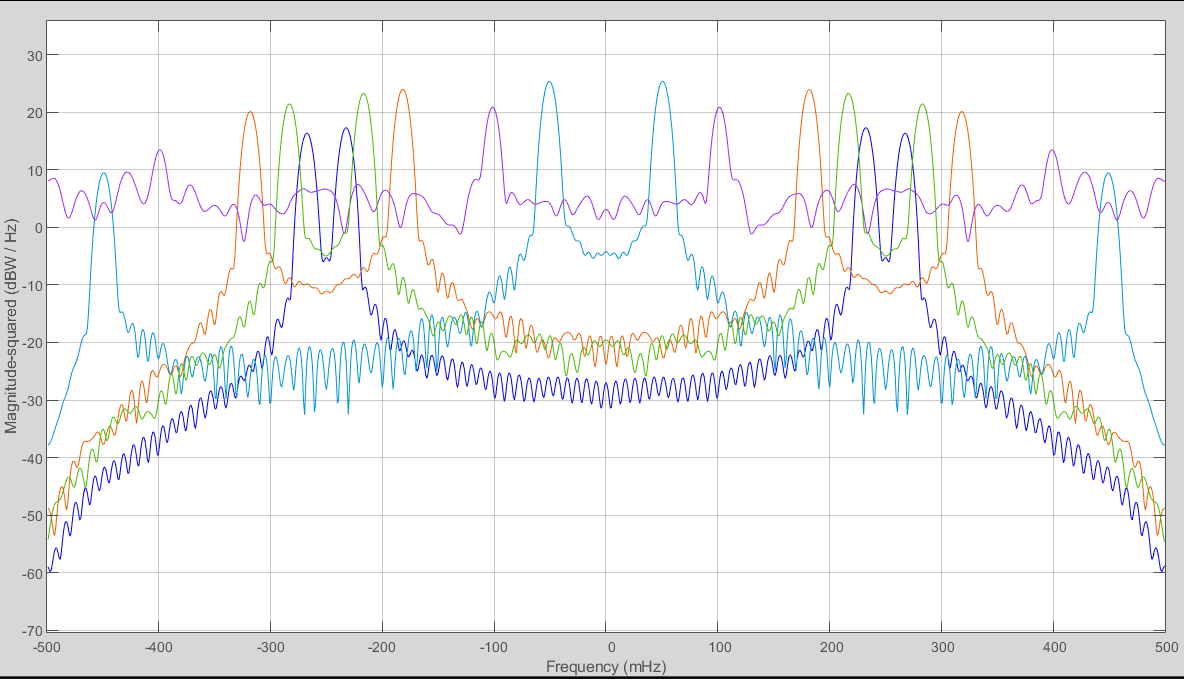
****

Fig. 3.11 Spectrum allocation of Primary Users (PU’s)

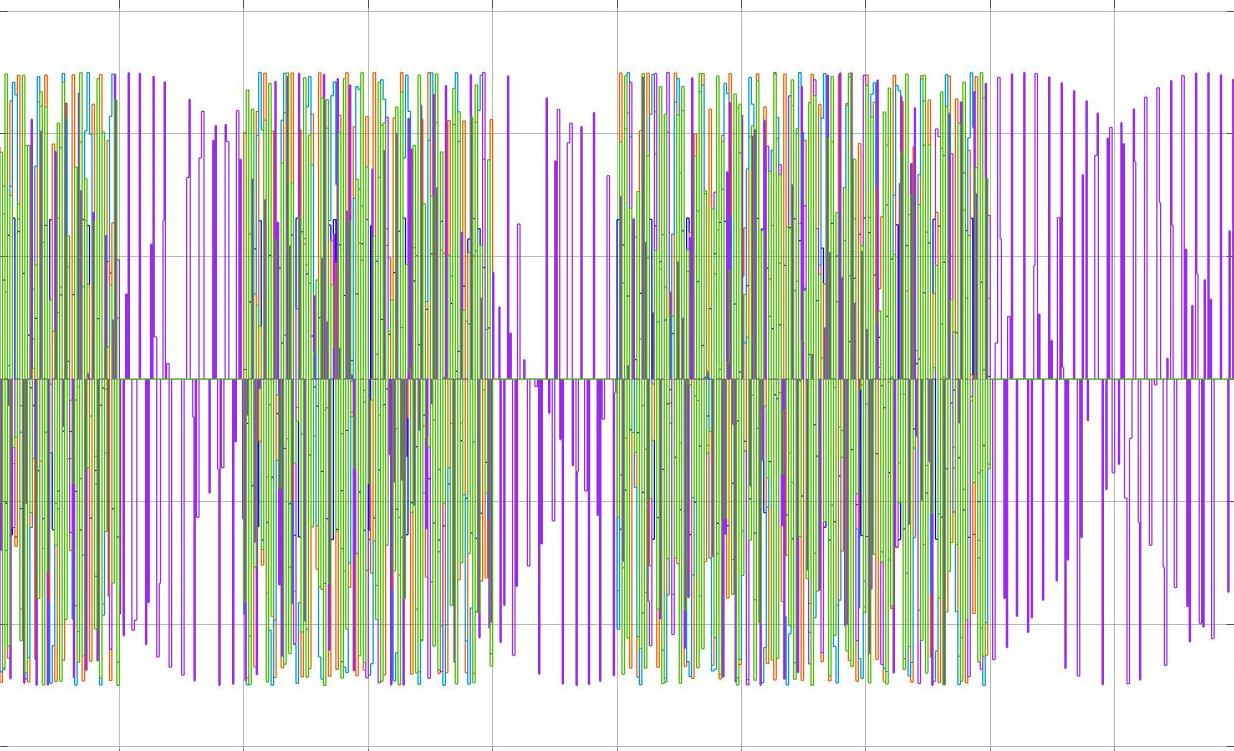


Fig. 3.11 Spectrum gaps while allocating Primary Users (PU’s)

## CHAPTER 4- NON-CSS TECHNIQUE (Single Threshold) USING MATLAB

**4.1 Energy Detection:**

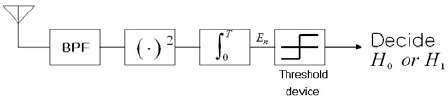
It is a non-coherent and non-cooperative detection method that detects the primary signal based on the sensed energy. Due to its simplicity and no requirement on a priori knowledge of primary user signal, energy detection (ED) is the most popular sensing technique in cooperative sensing.

Fig.4.1 Energy detector block diagram

The block diagram for the energy detection technique is shown in the **Figure 4.1** It is composed of four main blocks:

1. Band Pass Filter
2. Squaring Device
3. Integrator
4. Threshold device

In this method, signal is passed through band pass filter of the bandwidth W, then multiplied by itself (squaring device) and is integrated over time interval. The output from the integrator block is then compared to a predefined threshold. This comparison is used to discover the existence of absence of the primary user. The threshold value can set to be fixed or variable based on the channel conditions. The ED is said to be the Blind signal detector because it ignores the structure of the signal. It estimates the presence of the signal by comparing the energy received with a known threshold Vt derived from the statistics of the noise. Analytically, signal detection can be reduced to a simple identification problem, formalized as a Binary Hypothesis Testing Problem.

**4.1.1 Binary Hypothesis Testing Problem**

Depending on the idle state or busy state of the primary user, with the presence of the noise, the signal detection at the secondary user can be modeled as a Binary Hypothesis Testing Problem, given as:

Hypothesis 0 (H0): signal is absent

Hypothesis 1 (H1): signal is present

If the received signal, **y,** is sampled, the nth (n= 1, 2, 3 ) sample, y(n) can be given as:

*y(n)* = w(n)……H0

*y(n)* = *x(n)* + w(n)……H1

where x(n) is the signal transmitted by the PU, x(n)= h s(n) where his channel gain and w(n) is the noise sample which is assumed to be Gaussian random variable with mean zero

(IE[w(n)] = 0) and variance *2a* i.e., w(n) - *N (O,* 2a).

Then a **decision rule** can be stated as:





where *E* is the test statistic. Energy detection differentiates between the two hypotheses H0 and H1 by comparing *E* with threshold voltage *Vt* as shown*.* Setting the right threshold value is of critical importance. The key problem in this regard is illustrated in **Fig. 4.2,** which shows probability density functions of received signal with and without active PU.

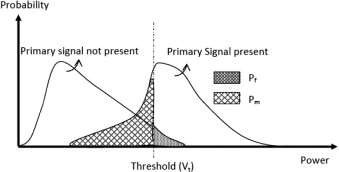


Fig. 4.2 Threshold setting in ED: trade-off between missed detection and false alarm

Hence if the selected *Vt* is too low, the false alarm probability i.e.,

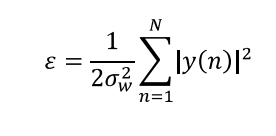
*Pf* = *Pr (ε* > *Vt | H0)* increases, which results in low spectrum utilization. On the other hand, if *Vt* is kept unnecessarily high, the probability of missed detection

*Pm= Pr (ε < Vt | H1)* is increased which may result in interference with an active PU. Hence, a careful tradeoff is considered while setting the threshold for ED.

**4.1.2 Test Statistic**

The output of the integrator (Fig. 4.2) is called ***decision (test) statistic.*** The test statistic is compared with the threshold to make the final decision on the presence/absence of the primary signal. However, the test statistic may not always be the integrator output, but a function that is monotonic with the integrator output

The test statistic of the energy detector can be given as



where *2σw2 is* the noise variance, N is sample number such that N=2TW, where TW is the time-bandwidth product.

The **performance of energy detector** is characterized by using following metrics, which have been introduced based on the test statistic under the binary hypothesis:

* **False alarm probability (Pf):** the probability of deciding the signal is present while H0 is true, i.e.



where *Vt* is the detection threshold, and P,[.] stands for an event probability. In the context of cognitive radio networks, a false alarm yields undetected spectrum holes. So, a large Pf contributes to poor spectrum usage by secondary users.

* **Missed-detection probability (Pmd):** The probability of deciding the signal is absent while H1 true is



which is equivalent to identifying a spectrum hole where there is none. Consequently, large Pmd introduces unexpected interference to primary users.

* **Detection probability (Pd)**: the probability of deciding the signal is present when H1 is true, i.e.,

*Pd= Pr(ε* > *Vc l H1)*



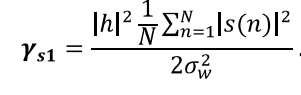
Both reliability and efficiency are expected from the spectrum sensing technique built into the cognitive radio, i.e., a higher Pct (or lower Pmd ) and lower Pf are preferred.

The statistical properties of *E* are necessary to characterize the performance of an energy detector. To get the statistical properties, signal and noise models are essential.

**4.1.3 Signal Models:**

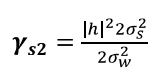
Based on the available knowledge of s(n) the receiver can adopt an appropriate model, which helps to analyze the distribution of the test statistic under H1. For example, three different models, **S1, S2** and **S3,** are popularly used in the literature, and are given as follows:

**S1:** For given channel gain **h,** the signal to be detected, y(n), can be assumed as Gaussian with mean *E[y(n)] = E [hs(n) + w(n)] = hs(n)* and variance 2σw2. This case may be modeled as an unknown deterministic signal. For the signal transmitted over a flat band-limited Gaussian noise channel, a basic mathematical model of the test statistic of an energy detector. The receive SNR can then be given as:

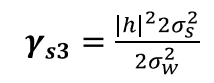


**S2:** When the receiver has very limited knowledge of the transmitted signal (e.g., signal distribution), the signal sample may be considered as Gaussian random variable, i.e., s(n) - *N (0,* 2σs2;) and then

*y(n)* - *N (0, 2(σw2 +σs2)).* The receiver SNR can then be given as



**S3:** If the Gaussian assumption is removed from **S2** signal model, and signal sample is considered as random variable with mean zero and variance 2σs2 but with an unknown distribution, then y(n) has mean zero and variance *2(σw2 +σs2 ).* The receiver SNR can also be given as



For a sufficiently large number of samples, the signal variance can be written as:



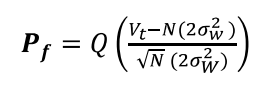
and thus, all the receive SNRs given in equation under different signal models have the same expression. In this case, the instantaneous SNR is denoted as *y.*

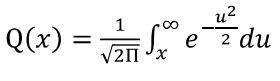
**4.1.4 Distribution of Test Statistics**

The exact distributions of test statistics for different signal models are analyzed in the following under both hypotheses, **H0** and H1• By CLT approach

*Under H0*

The false alarm probability can be given as:

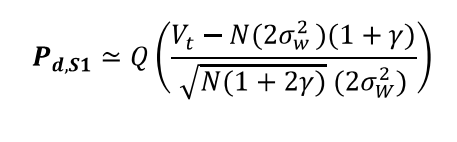
****

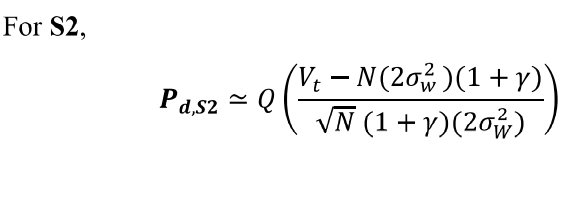


where is the Gaussian-Q function

*Under* H1

The probability detection pd for S1 is given as:





Note that *Pd, S3 has* the same expression as *Pd, s2·*

**4.2 Design Parameters:**

The main design parameters of the energy detector are the number of samples and threshold. Although the performance of the energy detector depends on SNR and noise variance as well, designers have very limited control over them because these parameters depend on the behavior of the wireless channel.

* + 1. **Threshold**

A pre-defined threshold E is required to decide whether the target signal is absent or present. This threshold determines all performance metrics, Pd, Pf and Pmd. Since it varies from 0 to l , selection of operating threshold is important. The operating threshold thus can be determined based on the target value of the performance metric of interest. When the threshold increases (or decreases), both Pr and Pd decrease (or increase). For known N and σw2, the common practice of setting the threshold is based on a constant false alarm probability Pf e.g., Pf ≤ 0.1. The selected threshold based on Pf can be given as:



However, this threshold may not guarantee that the energy detector achieves the target detection probability (e.g., 0.9 specified in the IEEE 802.22 WRAN). Thus, threshold selection can be viewed as an optimization problem to balance the two conflicting objectives (i.e., maximize Pd while minimizing Pf).

**4.2.2 Number of Samples**

The number of samples (N) is also an important design parameter to achieve the requirements on detection and false alarm probabilities. For given false alarm probability Pf and detection probability Pd, the minimum number of samples required can be given as a function of SNR. By eliminating Vt from both Pf and Pd (here signal model **S1** is used as an example), N can be given as



which is not a function of the threshold. Due to the monotonically decreasing property of function Q-1 ( . ), it can be seen that the signal can be detected even in very low SNR region by increasing N, when the noise power is perfectly known. Since *N=tfs* where r is the sensing time and fs is the sampling frequency, the sensing time increases as N increases. This is a main drawback in spectrum sensing at low SNR because of the limitation on the maximal allowable sensing time (e.g., the IEEE 802.22 specifies that the sensing time should be less than 2 s). Therefore, the selection of N is also an optimization problem.

**4.3 Simulation Results:**

ROC plots. For Energy Detector based spectrum sensing

Pmd= Probability of missed detection

Pd = Probability of detection

Pf = Probability of false alarm

N = Number of samples

SNR= Signal to noise ratio

Detection probability (Pd), False alarm probability (Pf) and missed detection probability (Pmd) are the key measurement metrics that are used to analyses the performance of spectrum sensing techniques. The performance of an spectrum sensing technique is illustrated by the receiver operating characteristics (ROC) curve which is a plot of Pd versus Pf (or) Pf versus Pm. The performance of energy detector is analyzed using ROC (Receiver operating characteristics) curves. Monte-Carlo method is used for simulation

* + 1. **The plot of Probability of false alarm versus Probability of detection:**

The plot of Probability of false alarm versus Probability of detection is illustrated in **Figure.4.3.** Probability of false alarm (Pf) is on X-axis and probability of detection (Pd) is on Y-axis. In the simulation, study input random bit stream is multiplied by 1 MHz sinusoidal carrier signal to get 1 MHz BPSK modulated signal, which is transmitted in AWGN channel. The detection performance can be performed by varying the probability of false alarm from 0.01,0.02……1 and finding the probability of detection by using Monte Carlo simulation. Here the number of sample points taken is **N=I000** and **SNR**= **-12dB.**

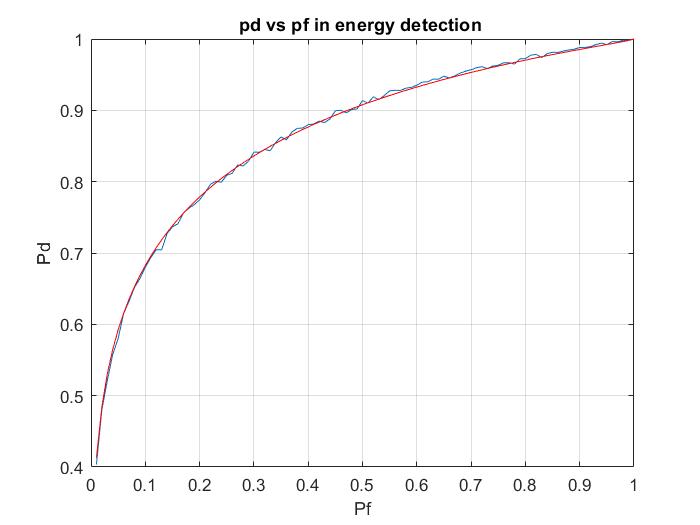


Fig.4.3 ROC curve for Pf vs Pd Energy detector-based spectrum sensing at SNR= - 12dB

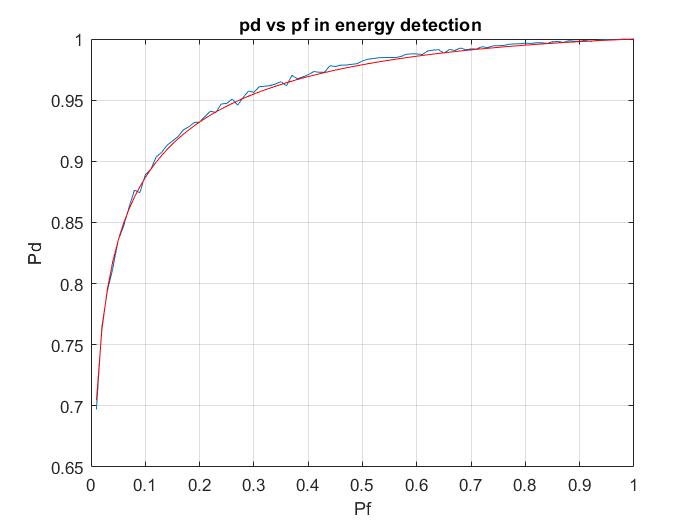


Fig.4.4 Plot between Pf and Pd at SNR= -l0db

It can be interpreted from Fig.4.3 that the performance of energy detector improves with increase in SNR and degrades with the decrease in SNR for the increase in probability of false alarm respectively. This is quantified in Table 4.1.

|  |  |  |  |
| --- | --- | --- | --- |
| **Probability of**  **False Alarm (Pf)** | **Probability of**  **Detection (Pd) (SNR = -12dB)** | **Probability of**  **Detection (Pd) (SNR= -l0dB)** | **Improvement in times** |
| 0.01 | 0.3 | 0.7 | 0.75 |
| 0.2 | 0.78 | 0.93 | 0.1923 |
| 0.4 | 0.87 | 0.97 | 0.115 |
| 0.6 | 0.94 | 0.98 | 0.425 |
| 0.8 | 0.97 | 0.99 | 0.0206 |
| 1.0 | 1.0 | 1.0 | 0 |

Table 4.1: Improvement inProbability of detection with increase in Signal to Noise Ratio in Energy Detection Method for AWGN Channel.

***RESULT:* Table 4.1** shows **that 2 dB increase** in Signal to Noise Ratio; increases the probability of detection (at SNR= -10dB) up to 0.75 times as compared to probability of detection (at SNR=-12dB) for AWGN Channel.

**4.3.2 The plot of Probability of detection (Pd) versus Signal to Noise Ratio (SNR):**

The plot of Probability of detection (Pd) and Signal to Noise Ratio (SNR) is illustrated in **Figure.4.5.** Signal to Noise Ratio (SNR) is on X-axis and probability of detection (Pd) is on Y-axis. The detection performance can be performed by varying the SNR from -20dB, 19dB….0dB and finding the probability of detection by using Monte Carlo simulation. Here ***PF* 0.01** and no. of samples **N.**

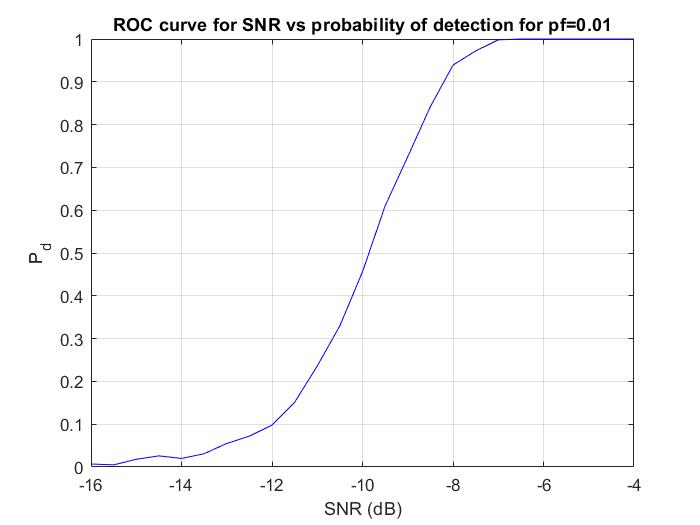
****

Fig.4.5 Plot for Pd vs SNR Energy detector-based spectrum sensing at N= 1000

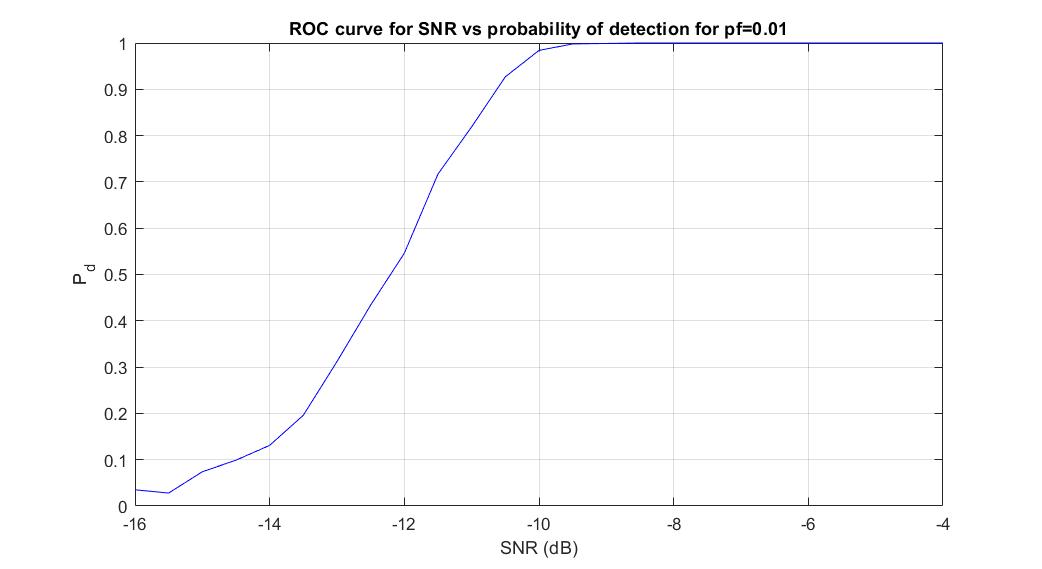
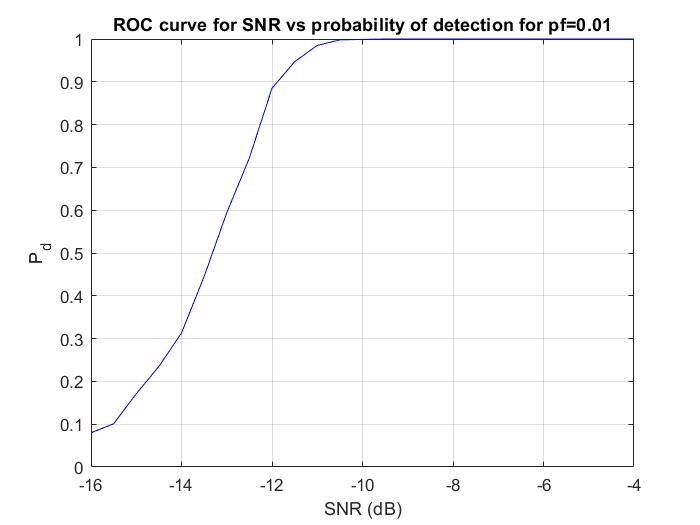
****

Fig.4.6 Plot for Pd vs SNR at N= 3000

The experiment is repeated for different number of sample points i.e., N= 3000, 5000 and 7000 and plotted in **fig.4.6, 4.7 and 4.8** respectively

****

F ig.4.7 Plot for Pd vs SNR at N= 5000

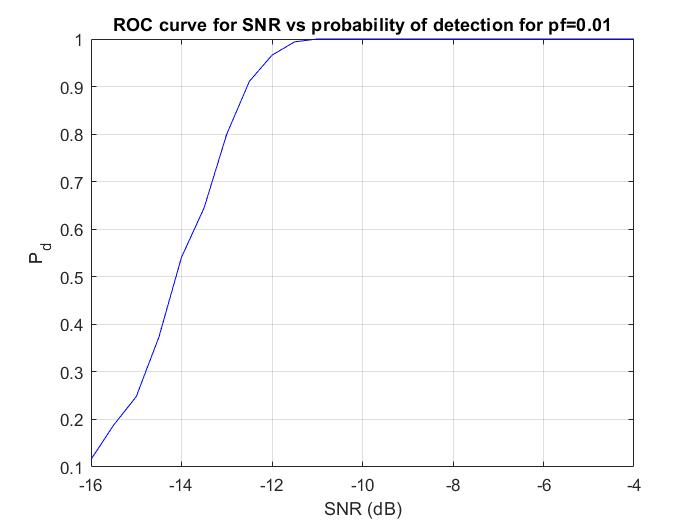
****

Fig. 4.8 Plot for Pd vs SNR at N= 7000

It can be interpreted from Fig. **4.6, 4.7 and 4.8** that the performance of energy detector improves with increase in the number of sample points. That is Pct increases with the increase in N for same values of SNR at fixed Pf. This effect is shown in **Table 4.2.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Signal**  **to Noise Ratio (SNR**  **in dB)** | **Probability of Detection (Pd)**  **(N= 1000)** | **Probability of Detection (Pd) (N=3000)** | **Probability of Detection (Pd) (N=5000)** | **Probability of Detection** (Pd) **(N=7000)** |
| -16 | 0.01 | 0.049 | 0.09 | 0.03 |
| -14 | 0.05 | 0.12 | 0.31 | 0.54 |
| -12 | 0.1 | 0.54 | 0.89 | 0.97 |
| -10 | 0.45 | 0.98 | 1 | 1 |
| -7 | 1 | 1 | 1 | 1 |

Table 4.2: Improvement in Probability of Detection with increase in number of samples in Energy Detection Method for AWGN Channel.

**RESULT: Table 4.2** shows that **increase** in the number of samples, increases the detection probability. For example, Pct increases from 0.55 to 0.89 when N is increased from 1000 to 7000. At SNR= -7dB, we get Pd=1.

* + 1. **The Plot of Threshold versus Probability of false alarm Pf:**

The plot of Threshold vs probability of false alarm (Pf) is illustrated in Figure.4.10. Threshold of energy detector is on X-axis and probability of false alarm (Pf) is on Y-axis. we calculate the threshold in energy detection by simulations This is a general method and applicable to all scenarios for energy detection. We assume that all the signals are complex Gaussian. Assume only noise is received, i.e., primary user is absent. If the only noise energy lies above the threshold, it corresponds to false alarm. **Probability of False Alarm = energy above threshold/No. of Iteration.** For **N=1000**

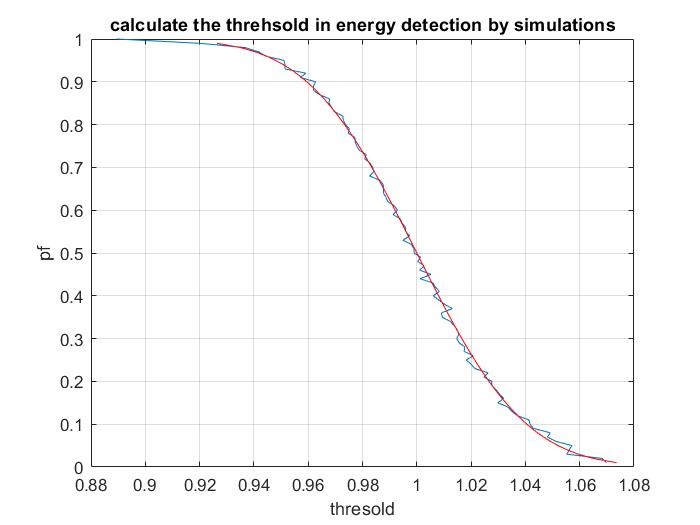
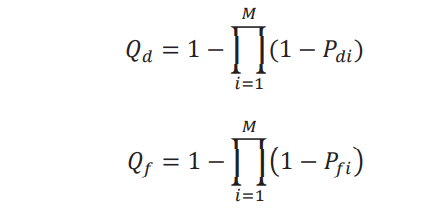


Figure.4.9: plot of Threshold vs probability of false alarm (Pf).

## CHAPTER 5-CSS TECHNIQUE (Double threshold) USING MATLAB

### **5.1 Cooperative Spectrum Sensing (CSS):**

To overcome the problem of node failure and fading, CSS was discussed earlier. CSS is illustrated in figure 5.1. In this diagram, each SU sends its local decision to the FC. FC then combines the result from each CR under certain fusion rules i.e., OR rule, AND rule and Majority rule. Probability of detection (Qd) and probability of false alarm (Qf) for the cooperative scheme under OR rule is given



Where Pd and Pf are the probability of detection and probability of false alarm of ith SU respectively and M is the number of CRs participating in CSS.

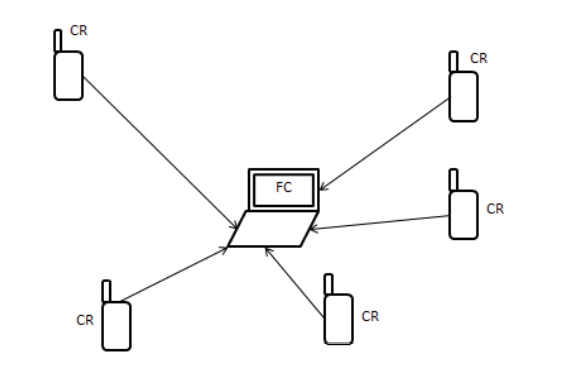
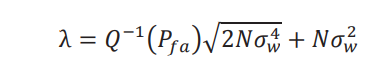


fig. 5.1. Cooperative spectrum sensing

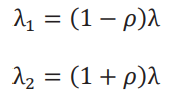
### **5.2 PROPOSED DOUBLE THRESHOLD ENERGY DETECTION IN COOPERATIVE SPECTRUM SENSING**

In conventional energy detector as already discussed, each CR makes their local decisions based on a single threshold as shown in figure 5.2.(a). If the energy calculated X (or observed energy Oi) is more than threshold λ then hypothesis H1 is true otherwise H0. Double threshold energy detection method can be illustrated with the help of figure 5.2.(b). Here hypothesis H1 is true if X is more than λ2 and hypothesis H0 is true when X is less than λ1 and in case observed energy X lies in between the two thresholds i.e., λ1<X<λ2, no decision is taken and CR will go for sensing again.

Threshold λ can be calculated



And thresholds λ1and λ2 can be found as:



Here ρ is the uncertainty parameter.

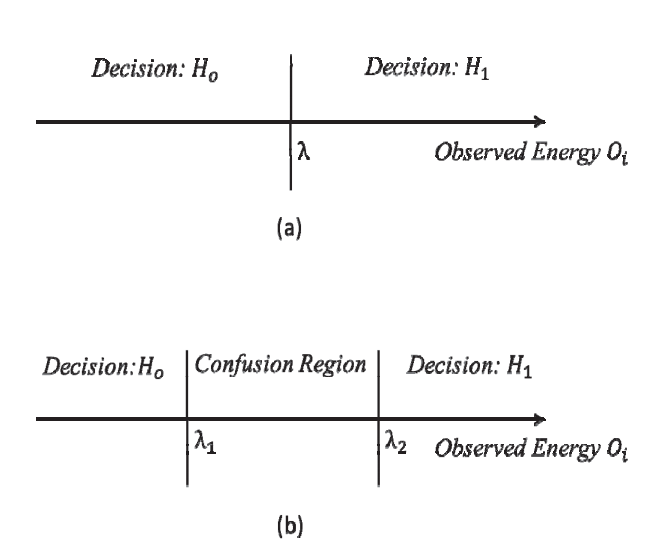


Fig. 5.2. (a) Conventional energy detector with a single threshold.

(b) Double threshold-based energy detector

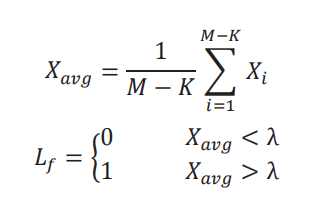
In this project, we have developed a simulation model for double threshold CSS which works as follows:

1. Each SU (i=1……M) will make **a** decision if and only if observed energy *Xi* lies above λ2 or below λ1 which is given by
   1. Xi € H0

Li = {

* 1. Xi € H1

1. If X lies between λ1 and λ2, then SU does not make a decision and report the observed energy *Xi* to the FC.
2. Let us assume that FC receives K local decisions and M-K energy values. FC will now take the average of all the energy values received and compare it with the threshold λ to make a decision *Ho* or *H1* as



* 1. 4. Now FC will be having K+1 decision: K decisions from the SUs which have made the decisions and one from the FC itself based on the observed energy received from the confused SUs.
  2. Now FC can combine these K+1 decision by OR fusion rule with the help of equation 6 and 7.

In this way, no sensing information is lost (every CR is making contribution to the final decision) and problem of sensing failure is also removed. Figure 4.3 shows the flowchart of the proposed scheme.

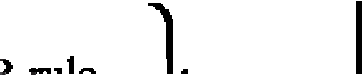
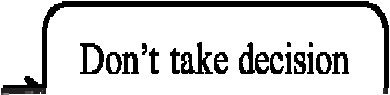
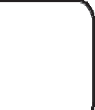
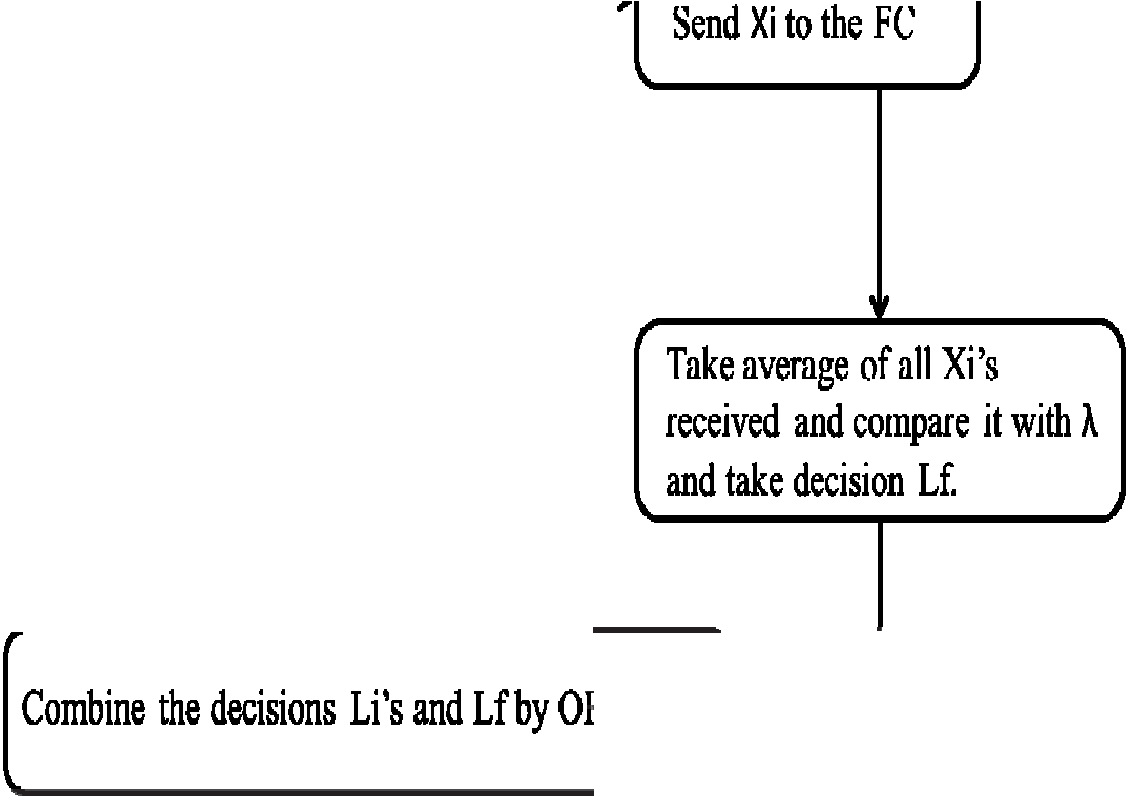
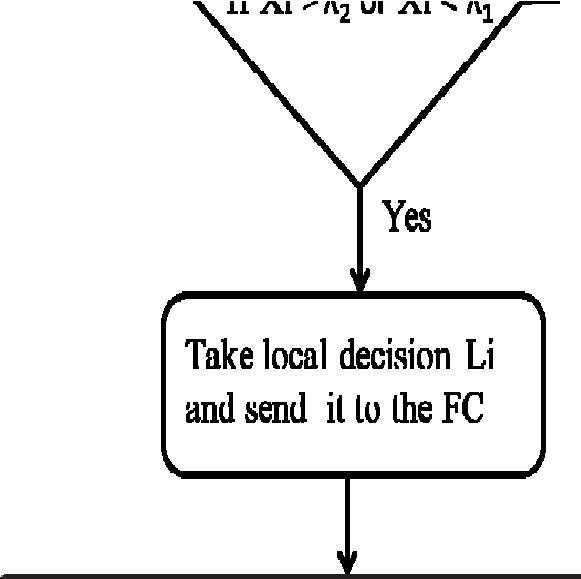
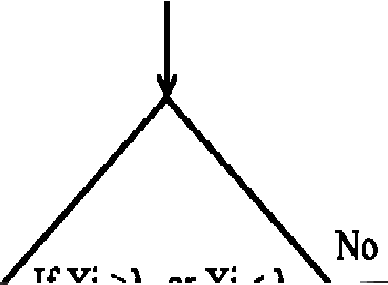
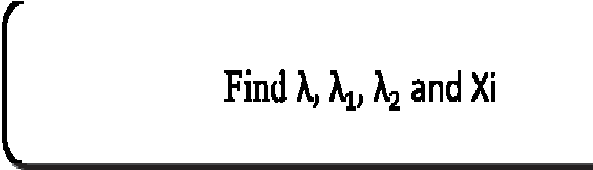


Fig. 5.3 Flow chart of the proposed scheme.

**5.3 Simulation results:**

In this section, we have performed simulation study to evaluate the performance of the proposed scheme. First, we developed a simulation model in MATLAB for a single threshold energy detection method and compared the results with the theoretical results to validate the simulation model Figure 5.4 shows the ROC (Receiver Operating characteristics) for theoretical and simulation model with N=500 & 1000. This shows that the developed simulation model is very close to the theoretical model. Then this simulation model is extended for the proposed scheme to evaluate the performance. We have assumed a BPSK modulated signal.

Figure 5.5 shows the ROC (curve between Pd vs Pfa) for single and double threshold at -5 dB SNR with 200 samples. For this, we have not considered the confused region. For simplicity, we have assumed that every CR is using the same value of threshold λ with uncertainty parameter, q= 0.1. Figure 5.6 shows the ROC for the conventional and proposed scheme. We have assumed the number of SUs participating in the cooperation, M=5. This figure clearly shows that the proposed scheme outperforms the conventional CSS. This curve is drawn at -8dB SNR with 100 samples. It has been clearly observed from the graph that proposed scheme is giving an almost 10% improvement in the probability of detection at Pfa=0.1.

Figure 5.7 shows the graph between Qd and SNR by fixing Pfa=0.1 and it has been observed from the graph that at low SNR, the proposed scheme performs better. Figure 5.8 shows the curve between probability of decision error (Pe) and SNR and it has been observed that the proposed scheme is minimizing the decision error in low SNR region.

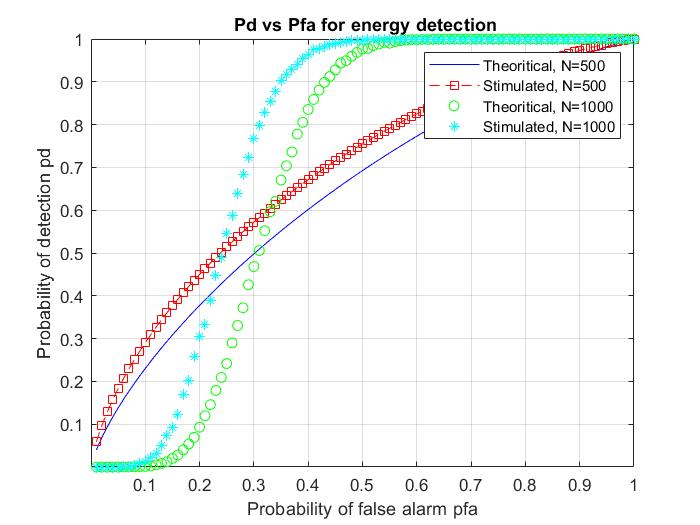
****

Fig. 5.4. Pd vs Pfa for energy detection

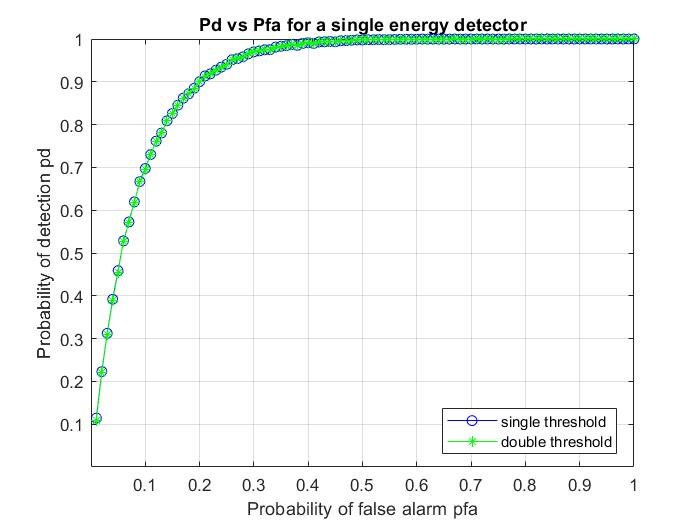


Fig. 5.5. Pd vs Pfa for a single energy detector

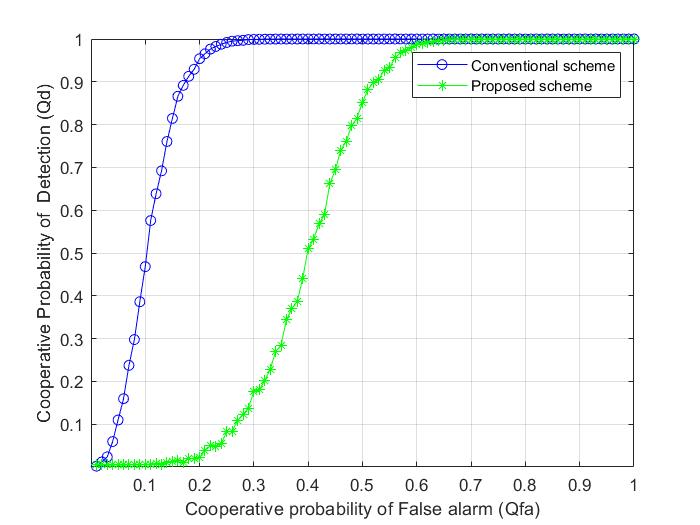
****

Fig. 5.6. Pd vs Pfa for conventional CSS and Proposed CSS.

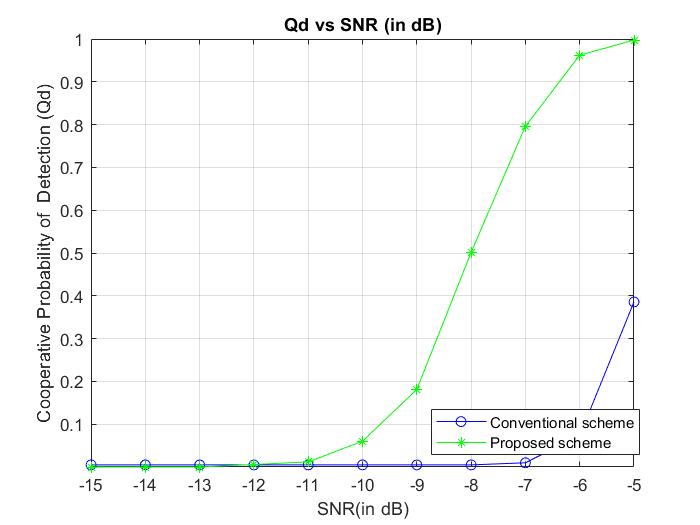
****

Fig. 5.7. Qd vs SNR (in dB)

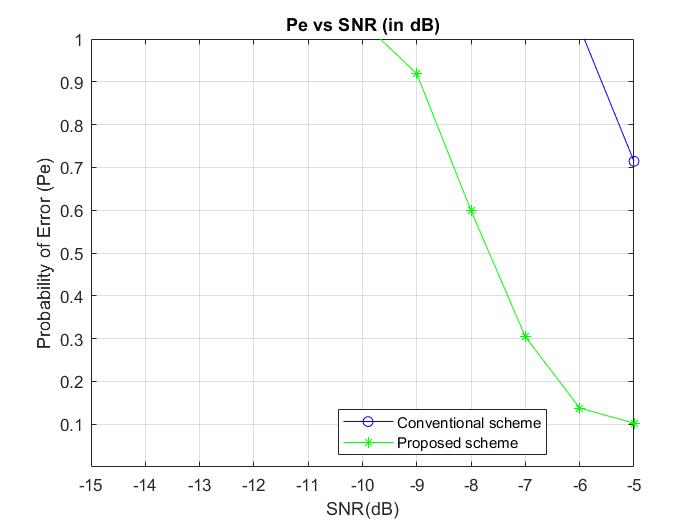
****

Fig. 5.8. Pe vs SNR (in dB)

**Result:**

FC receives two types of decisions i.e., local decisions and observed energy values. We have proposed a scheme under which FC averages the observed energy values and compare it with threshold λ to make a decision Lf. Then FC combines all the local decisions received from SUs and decision Lf by OR rule of fusion to make a global decision. It has been observed that there is a significant improvement in the detection performance by employing the proposed scheme. Moreover, the problem of sensing failure has also been removed. In future, optimization between the spectrum sensing performance and overhead burden because of sending the energy values to the FC can be explored.

## CHAPTER 6- CONCLUSION

Spectrum is a very valuable resource in wireless communication systems, and it has been a focal point for research and development efforts over the last several decades. Cognitive radio, which is one of the efforts to utilize the available spectrum more efficiently through opportunistic spectrum usage, has become an exciting and promising concept. In this paper, we examined various aspects of cognitive radio and identified spectrum sensing as the prerequisite requirement for the deployment of cognitive radio oriented wireless networks. In this thesis we came to know about how the Spectrum Allocation and Spectrum Sensing is taking part in modern Cognitive Radio Network.

*6.1 Spectrum Allocation:*

The approach was to take the decisions on the basis of power spectral density of the channel which can be used cognitively to search the available spectral gaps those can be used to new incoming users (SU) thus improving the overall channel’s throughput. In this work the energy detection spectrum sensing using FFT within the specified frequency band is performed. It has been shown that how the cognitive radio works dynamically with changing the frequency band from one to another and successfully demonstrated in simulation result. That is the Spectrum Access in Cognitive Radio demonstrated successfully without interfering with the other frequency bands used by the primary user (PU).

*6.2 Spectrum Sensing:*

studies have been carried out on spectrum sensing based on energy detection in Cognitive Radio Networks. Simulations were carried out and graphs of probability of detection vs. the probability of false alarm were observed and analyzed. The detection probability increases with respect to the increase in false alarm. Significant reduction in probability of missed detection have been achieved with this sensing technique as evidenced from the simulation results. The detection probability also varies with the SNR value. SNR has a great influence on the probability of detection. With an increase in SNR value, the probability of detection increases. Hence, we almost obtain the final result on energy detection according to our expectation. By using the previous Energy detector, we implemented the concept in co-operative spectrum sensing for better performance. Hybrid Spectrum Sensing techniques like any two combinations of spectrum sensing techniques such as Energy detector and Matched detector (or) Cyclostationary detector, with Improved Double threshold Energy detection in both Co-operative, non-cooperative sensing needed for better detection performance. It is well-known that energy detector’s performance is susceptible to uncertainty in noise power under such cases alternate detection schemes such as Cyclic feature detection.

By this thesis, we conclude that the double threshold cooperative spectrum sensing is based on the conventional energy detection technique is the best technique where it can reduce the total error rate by finding 2 thresholds in the conventional energy detection.

**CHAPTER 7- REFERENCES**

1. Mitola, Joseph (2000), ["Cognitive Radio](http://kth.diva-portal.org/smash/record.jsf?pid=diva2:8730) - [An Integrated Agent Architecture for Software Defined Radio"](http://kth.diva-portal.org/smash/record.jsf?pid=diva2:8730), *Diva* (Ph.D. Dissertation), Kista, Sweden: [KTH Royal Institute of Technology](https://en.wikipedia.org/wiki/Royal_Institute_of_Technology), [ISSN](https://en.wikipedia.org/wiki/ISSN_(identifier)) [1403-5286](https://www.worldcat.org/issn/1403-5286)
2. V. Valenta et al., ["Survey on spectrum utilization in Europe: Measurements, analyses and observations"](http://hal.inria.fr/docs/00/49/20/21/PDF/paper9220_valenta.pdf), Proceedings of the Fifth International Conference on Cognitive Radio Oriented Wireless Networks & Communications (CROWNCOM), 2010
3. ["Cognitive Functionality in Next Generation Wireless Networks"](https://web.archive.org/web/20081118234117/http:/grouper.ieee.org/groups/scc41/files/Communications_Magazine_article_on_SCC41.pdf)(PDF)*. Archived from*[the original](http://grouper.ieee.org/groups/scc41/files/Communications_Magazine_article_on_SCC41.pdf)(PDF)*on 18 November 2008*. Retrieved 6 June 2009*.*
4. H. Sun, A. Nallanathan, C.-X. Wang, and Y.-F. Chen, ["Wideband spectrum sensing for cognitive radio networks: a survey"](http://ieeexplore.ieee.org/xpl/articleDetails.jsp?reload=true&arnumber=6507397), IEEE Wireless Communications, vol. 20, no. 2, pp. 74–81, April 2013
5. Carlos Cordeiro, Kiran Challapali, and Dagnachew Birru. Sai Shankar N. IEEE 802.22: An Introduction to the First Wireless Standard based on Cognitive Radios JOURNAL OF COMMUNICATIONS, VOL. 1, NO. 1, APRIL 2006
6. Z. Li, F.R. Yu, and M. Huang, ["A Distributed Consensus-Based Cooperative Spectrum Sensing in Cognitive Radios"](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5229125), IEEE Trans. Vehicular Technology, vol. 59, no. 1, pp. 383-393, Jan. 2010.
7. H. Uchiyama , K. Umebayashi , Y. Kamiya , Y. Suzuki , T. Fujii , F. Ono and K. Sakaguchi "Study on cooperative sensing in cognitive radio based ad-hoc network", Proc. IEEE PIMRC 2007, pp.1 -5 2007.
8. S. Zarrin and T. J. Lim "Belief propagation on factor graphs for cooperative spectrum sensing in cognitive radio", Proc. IEEE DySPAN 2008, pp.1 -9 2008
9. E. Visotsky , S. Kuffner and R. Peterson "On collaborative detection of TV transmissions in support of dynamic spectrum sharing",
10. C. R. Stevenson , G. Chouinard , Z. Lei , W. Hu , S. J. Shellhammer and W. Caldwell "IEEE 802.22: The first cognitive radio wireless regional area network standard", IEEE Commun. Mag., vol. 47, no. 1, pp.130 -138 2009
11. P. Verma and B. Singh, "Simulation study of double threshold energy detection method for cognitive radios," 2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN), 2015, pp. 232-236, doi: 10.1109/SPIN.2015.7095276.

# APPENDIX

## MATLAB CODES:

### **1. Spectrum allocation MATLAB code:**

clc

close all

clear all

t = 0:0.00001:0.001;

F1 = 1000;

F2 = 2000;

F3 = 3000;

F4 = 4000;

F5 = 5000;

Fs = 12000;

a1 = 1; a2 = 0; a3 = 0; a4 = 0; a5 = 0; A = 0; a = 0;

x = cos(2\*pi\*1000\*t);

in\_p = input('\nDo you want to enter first primary user Y/N: ','s');

if(in\_p == 'Y' | in\_p == 'y')

a1 = ammod(x,F1,Fs);

end

in\_p = input('Do you want to enter second primary user Y/N: ','s');

if(in\_p == 'Y' | in\_p == 'y')

a2 = ammod(x,F2,Fs);

end

in\_p = input('Do you want to enter third primary user Y/N: ','s');

if(in\_p == 'Y' | in\_p == 'y')

a3 = ammod(x,F3,Fs);

end

in\_p = input('Do you want to enter fourth primary user Y/N: ','s');

if(in\_p == 'Y' | in\_p == 'y')

a4 = ammod(x,F4,Fs);

end

in\_p = input('Do you want to enter fifth primary user Y/N: ','s');

if(in\_p == 'Y' | in\_p == 'y')

a5 = ammod(x,F5,Fs);

end

a = a1 + a2 + a3 + a4 + a5;

while(1)

Pxx = periodogram(a);

Hpsd = dspdata.psd(Pxx,'Fs',Fs);

plot(Hpsd);

in\_p = input('\nDo you want to enter a secondary user Y/N: ','s');

if(in\_p == 'Y' | in\_p == 'y')

tp=0;

c1 = Pxx(25)\*10000 ;

c2 = Pxx(46)\*10000 ;

c3 = Pxx(62)\*10000 ;

c4 = Pxx(89)\*10000 ;

c5 = Pxx(105)\*10000 ;

gamma=8000;

if(c1 < gamma)

disp('Assigned to User 1 as it was not present.');

a1 = ammod(x,F1,Fs);

elseif (c2 < gamma)

disp('Assigned to User 2 as it was not present.');

a2 = ammod(x,F2,Fs);

elseif(c3 < gamma)

disp('Assigned to User 3 as it was not present.');

a3 = ammod(x,F3,Fs);

elseif(c4 < gamma)

disp('Assigned to User 4 as it was not present.');

a4 = ammod(x,F4,Fs);

elseif(c5 < gamma)

disp('Assigned to User 5 as it was not present.');

a5 = ammod(x,F5,Fs);

else

disp('all user slots in use. try again later,');

tp=1;

end

figure

a = a1 + a2 + a3 + a4 + a5 ;

Pxx = periodogram(a);

Hpsd = dspdata.psd(Pxx,'Fs',Fs);

plot(Hpsd);

%,,,,,,,,,,,slot empty operation,,,,,,,,,,,,,,

if(tp==1)

inp\_t=input('do u want to empty a slot: ','s');

if(inp\_t=='Y'|inp\_t=='y')

inp\_t=input('which slot do u want to empty for ur entry: ','s');

switch(inp\_t)

case ('1')

a1=0;

disp('slot1 is fired');

a = a1 + a2 + a3 + a4 + a5;

Pxx = periodogram(a);

Hpsd = dspdata.psd(Pxx,'Fs',Fs);

plot(Hpsd);

%break;

case('2')

a2=0;

disp('slot2 is fired');

a = a1 + a2 + a3 + a4 + a5;

Pxx = periodogram(a);

Hpsd = dspdata.psd(Pxx,'Fs',Fs);

plot(Hpsd);

%break;

case('3')

a3=0;

disp('slot3 is fired');

a = a1 + a2 + a3 + a4 + a5;

Pxx = periodogram(a);

Hpsd = dspdata.psd(Pxx,'Fs',Fs);

plot(Hpsd);

%break;

case('4')

a4=0;

disp('slot4 is fired');

a = a1 + a2 + a3 + a4 + a5;

Pxx = periodogram(a);

Hpsd = dspdata.psd(Pxx,'Fs',Fs);

plot(Hpsd);

%break;

case('5')

a5=0;

disp('slot5 is fired');

a = a1 + a2 + a3 + a4 + a5;

Pxx = periodogram(a);

Hpsd = dspdata.psd(Pxx,'Fs',Fs);

plot(Hpsd);

%break;

otherwise disp('invalid slot entered');

%break;

end %switch end

end

end

%////////////////////Adding Noise////////////////////

inp\_t=input('do u want to add noise(y / n): ','s');

if(inp\_t=='y'|inp\_t=='Y')

d = input('Enter the SNR in dB: ');

figure

A = awgn(a,d);

Pxx1 = periodogram(A);

Hpsd = dspdata.psd(Pxx1,'Fs',Fs);

plot(Hpsd);

disp('adding noise');

c1 = Pxx1(25).\*10000;

c2 = Pxx1(49).\*10000;

c3 = Pxx1(62).\*10000;

c4 = Pxx1(89).\*10000;

c5 = Pxx1(105).\*10000;

if(c1 < gamma)

disp('User 1 is not present.');

else

disp('User 1 is present.');

end

if(c2 < gamma)

disp('User 2 is not present.');

else

disp('User 2 is present.');

end

if(c3 < gamma)

disp('User 3 is not present.');

else

disp('User 3 is present.');

end

if(c4 < gamma)

disp('User 4 is not present.');

else

disp('User 4 is present.');

end

if(c5 < gamma)

disp('User 5 is not present.');

else

disp('User 5 is present.');

end

end

end

%if rerun the program

temp = input('Do you want to re-run the program? [Y/N]: ','s');

if(temp == 'Y' | temp == 'y')

disp('\n\nEnter the users again.\n\n');

else

break;

end

end

### **2.Non-Cooperative Spectrum Sensing (Single Threshold) MATLAB code:**

### **2.1 Plot of Probability of false alarm vs Probability of detection:**

clc

close all

clear all

L = 1000;

snr\_dB = input('Enter the value of SNR(neg): '); % SNR in decibels

snr = 10.^(snr\_dB./10); % Linear Value of SNR

Pf = 0.01:0.01:1; % Pf = Probability of False Alarm

%% Simulation to plot Probability of Detection (Pd) vs. Probability of False Alarm (Pf)

for m = 1:length(Pf)

m

i = 0;

for kk=1:10000 % Number of Monte Carlo Simulations

n = randn(1,L); %AWGN noise with mean 0 and variance 1

s = sqrt(snr).\*randn(1,L); % Real valued Gaussina Primary User Signal

y = s + n; % Received signal at SU

energy = abs(y).^2; % Energy of received signal over N samples

energy\_fin =(1/L).\*sum(energy); % Test Statistic for the energy detection

thresh(m) = (qfuncinv(Pf(m))./sqrt(L))+ 1; % Theoretical value of Threshold, refer, Sensing Throughput Tradeoff in Cognitive Radio, Y. C. Liang

if(energy\_fin >= thresh(m)) % Check whether the received energy is greater than threshold, if so, increment Pd (Probability of detection) counter by 1

i = i+1;

end

end

Pd(m) = i/kk;

end

plot(Pf, Pd);

xlabel('Pf');

ylabel('Pd');

hold on

%% Theroretical ecpression of Probability of Detection; refer above reference.

thresh = (qfuncinv(Pf)./sqrt(L))+ 1;

Pd\_the = qfunc(((thresh - (snr + 1)).\*sqrt(L))./(sqrt(2).\*(snr + 1)));

plot(Pf, Pd\_the, 'r');

hold on

### **2.2 Plot of SNR vs Probability of detection:**

clc;

clear all;

snrdb = -16:0.5:-4;

%%

% We are using BPSK modulation for test.

M = 2; % modulation index for psk

bpsk = comm.PSKModulator('ModulationOrder',M,...

'BitInput',false,...

'PhaseOffset',0); % M-psk modulator

nSample = input('enter the number of samples n: '); % samples in signal

pde = zeros(1,numel(snrdb)); % array for Pd

L = numel(snrdb);

% Loop for SNR

bWait = waitbar(0,'please wait...');

for i = 1:L % for all snr values

d = 0; % detection counter set to zeros

% Loop for 1000 tests

for j = 1:1000 % 1000 simulations

infoSignal = randi(M,nSample,1)-1; % random binary signal (bits = log2(M))

txSignal = step(bpsk,infoSignal); % M-psk signal

rxSignal = awgn(txSignal,snrdb(i)); % AWGN channel

pf = 0.01; % probabity of false detection

snr = 10^(snrdb(i)/20);

nvar = 1/snr; % noise variance

thresh = sqrt(2\*nSample\*nvar^4)\*qfuncinv(pf)+nSample\*nvar^2; % threshold value

energy = sum(abs(rxSignal).^2); % energy of signal

if energy > thresh % if energy is greater than threshold then signal is present

d = d+1;

end

end

pde(i) = d/1000; % avg over 1000 simulation

waitbar(i/L,bWait);

end

close(bWait);

%%

% Plot result (SNR Vs Pd)

figure()

plot(snrdb,pde,'b');

xlabel('SNR (dB)');

ylabel('P\_d');

title('ROC curve for SNR vs probability of detection for pf=0.01');

grid on;

#### 2.3 Plot of threshold vs Probability of false alarm:

clc

close all

clear all

L = 1000; % Number of samples to be taken

iter = input('enter the number of iterations: '); % Number of iterations

Pf = 0.01:0.01:1; % Probability of False Alarm

for tt = 1:length(Pf)

for kk=1:iter % Number of Monte Carlo Simulations

n=(randn(1,L)+j\*randn(1,L))./(sqrt(2)); % Primary User Gaussian Signal

y = n; % Received signal at the secondary user

energy = abs(y).^2; % Energy of received signal over L samples

energy\_fin(kk) =(1/L).\*sum(energy); % Test Statistic of the energy detection

end

energy\_desc = sort(energy\_fin,'descend'); % Arrange values in descending order

thresh(tt) = energy\_desc(ceil(Pf(tt)\*iter)); % Threshold obtained by simulations; the first 'Pf' fraction of values lie above the threshold

end

plot(thresh, Pf);

xlabel('thresold');

ylabel('pf');

grid on;

hold on;

thresh1 = (qfuncinv(Pf)./sqrt(L))+ 1; % Theroretical value of threshold

plot(thresh1, Pf, 'r');

xlabel('thresold');

ylabel('pf');

title("calculate the threhsold in energy detection ");

grid on;

hold on;

### **3.Cooperative Spectrum Sensing (Double Threshold) MATLAB code:**

### **3.1 Pd vs Pfa for a single energy detector MATLAB code:**

clc;

clear all;

format long

N = [500,1000];

snr\_dB =-9; %dB

snr = 10.^(snr\_dB./10);

Pf = 1/100:1/100:1;

%-------------------------------------

%% BPSK Signal

L=1500;

data = round(rand(1,L)); % Data sequence

uni2bip=2\*data-1; % Convert unipolar to bipolar

T=1; % Bit duration

Eb=T/2; % This will result in unit amplitude waveforms

fc=3/T; % Carrier frequency

t=linspace(0,5,1500); % discrete time sequence between 0 and 5\*T (15000 samples)

K=length(t); % Number of samples

Nsb=K/length(data); % Number of samples per bit

dd=repmat(data',1,Nsb); % replicate each bit Nsb times

bb=repmat(uni2bip',1,Nsb); dw=dd'; % Transpose the rows and columns

dw=dw(:)';

%------ Convert dw to a column vector (colum by column) and convert to a row vector

bw=bb';

bw=bw(:)'; % Data sequence samples

w=sqrt(2\*Eb/T)\*cos(2\*pi\*fc\*t); % carrier waveform

bpsk\_w=bw.\*w; % modulated waveform

%-----AWGN noise with mean 0 and variance -----%

Noise1 = randn(1,N(1));

Noise2 = randn(1,N(2));

vn1=var(Noise1);

vn2=var(Noise2);

%-----Real valued Gaussian Primary User Signal------%

Signal1 = sqrt(snr).\*bpsk\_w(1:500);

Signal2 = sqrt(snr).\*bpsk\_w(1:1000);

vs1=var(Signal1);

vs2=var(Signal2);

%------- Threshold-----------

Threshold\_1 = N(1)\*vn1 + qfuncinv(Pf)\*sqrt(2\*N(1)\*(vn1)^2);

Threshold\_2 = N(2)\*vn2 + qfuncinv(Pf)\*sqrt(2\*N(2)\*(vn2)^2);

%------------------------------------

%% Probabilty of detection theory

Pd\_the1 = qfunc((Threshold\_1 -N(1)\*(vn1+vs1))./(sqrt(2\*N(1)\*(vn1+vs1)^2)));

Pd\_the2 = qfunc((Threshold\_2 -N(2)\*(vn2+vs2))./(sqrt(2\*N(2)\*(vn2+vs2)^2)));

%% Probabilty of detection simulated

hwait = waitbar(0,'Please wait ....');

for i=1:length(Pf)

D1=0;

D2=0;

for j=1:10000 %Monte Carlo simulation

%-----AWGN noise with mean 0 and variance 1-----%

Noise1 = randn(1,N(1));

Noise2 = randn(1,N(2));

v\_n1=var(Noise1);

v\_n2=var(Noise2);

%-----BPSK Signal ------%

Signal1 = sqrt(snr).\*bpsk\_w(1:500);

Signal2 = sqrt(snr).\*bpsk\_w(1:1000);

v\_s1=var(Signal1);

v\_s2=var(Signal2);

Recv\_Sig1 = Signal1 + Noise1; % Received signal at SU 1

Recv\_Sig2 = Signal2 + Noise2; % Received signal at SU 2

Energy1 = abs(Recv\_Sig1).^2; % Energy of received signal over N samples

Energy2 = abs(Recv\_Sig2).^2;

%-----Computation of Test statistic for energy detection-----%

Test\_Statistic1 =sum(Energy1);

Test\_Statistic2 =sum(Energy2);

%-----Theoretical value of Threshold-----%

Threshold1(i) = N(1)\*v\_n1 + qfuncinv(Pf(i))\*sqrt(2\*N(1)\*(v\_n1)^2);

Threshold2(i) = N(2)\*v\_n2 + qfuncinv(Pf(i))\*sqrt(2\*N(2)\*(v\_n2)^2);

if(Test\_Statistic1>= Threshold1(i)) % Check whether the received energy is greater than threshold, if so,(Probability of detection) counter by 1

D1 = D1+1;

end

if(Test\_Statistic2 >= Threshold2(i)) % Check whether the received energy is greater than threshold, if so,(Probability of detection) counter by 1

D2 = D2+1;

end

end

Pd\_1(i)=D1/j;

Pd\_2(i)=D2/j;

waitbar(i/length(Pf),hwait);

end

close(hwait);

plot(Pf,Pd\_the1,'b')

grid on

hold on

plot(Pf,Pd\_the2,'r--s');

plot(Pf,Pd\_1,'g o');

plot(Pf,Pd\_2,'c \*');

ylabel("Probability of detection pd");

xlabel("Probability of false alarm pfa");

axis([0.0001,1,0.0001,1]);

legend('Theoritical, N=500','Stimulated, N=500',...

'Theoritical, N=1000','Stimulated, N=1000')

### **3.2 Pd vs Pfa for a single energy detector MATLAB code:**

clc;

clear all;

format long

N = 200;

snr\_dB =-5; %dB

snr = 10.^(snr\_dB./10);

Pf = 0.01:0.01:1;

uc = 0.1;

%% BPSK Signal

L=1500;

data = round(rand(1,L)); % Data sequence

uni2bip=2\*data-1; % Convert unipolar to bipolar

T=1; % Bit duration

Eb=T/2; % This will result in unit amplitude waveforms

fc=3/T; % Carrier frequency

t=linspace(0,5,1500); % discrete time sequence between 0 and 5\*T (15000 samples)

K=length(t); % Number of samples

Nsb=K/length(data); % Number of samples per bit

dd=repmat(data',1,Nsb); % replicate each bit Nsb times

bb=repmat(uni2bip',1,Nsb); dw=dd'; % Transpose the rows and columns

dw=dw(:)';

%------ Convert dw to a column vector (colum by column) and convert to a row vector

bw=bb';

bw=bw(:)'; % Data sequence samples

w=sqrt(2\*Eb/T)\*cos(2\*pi\*fc\*t); % carrier waveform

bpsk\_w=bw.\*w; % modulated waveform

%% Probabilty of single and double threshold

Pf2 = Pf;

hwait = waitbar(0,'Please wait....');

for i =1:length(Pf)

D\_sg=0;

D\_db=0;

for j=1:10000

%-----AWGN noise with mean 0 and variance -----%

Noise = randn(1,N);

vn=var(Noise);

%-----Real valued Gaussian Primary User Signal------%

Signal = sqrt(snr).\*bpsk\_w(1:200);

vs=var(Signal);

Recv\_Sig = Signal + Noise; % Received signal at SU 1

Energy = abs(Recv\_Sig).^2; % Energy of received signal over N samples

%------- Threshold-----------

Threshold\_0(i) = N\*vn + qfuncinv(Pf(i))\*sqrt(2\*N\*vn^2);

Threshold\_1(i) = (1-uc)\*Threshold\_0(i);

Threshold\_2(i) = (1+uc)\*Threshold\_0(i);

%------------------------------------

%-----Computation of Test statistic for energy detection-----%

X =sum(Energy);

%---------------------------------------

if ( X > Threshold\_0(i) )

%Pd2(i) = qfunc((Threshold\_0(i) -N\*(vn+vs))./(sqrt(2\*N\*(vn+vs)^2)));

D\_sg = D\_sg +1;

end

if ( X > Threshold\_2(i) )

%Pd2(i) = qfunc((Threshold\_0(i) -N\*(vn+vs))./(sqrt(2\*N\*(vn+vs)^2)));

D\_db = D\_sg +1;

end

end

Pd\_sg(i) = D\_sg/j ;

Pd2(i) = D\_db/j ;

waitbar(i/length(Pf),hwait);

end

close(hwait);

plot(Pf,Pd\_sg,'b-o');

grid on

hold on

plot(Pf,Pd2,'g-\*');

ylabel("Probability of detection pd");

xlabel("Probability of false alarm pfa");

axis([0.0001,1,0.0001,1]);

legend('single threshold','double threshold');

### **3.3 Cooperative probability of False alarm (Qfa) vs Cooperative Probability of Detection (Qd) MATLAB code:**

clc;

clear all;

format long

N = 200;

snr\_dB =-8; %dB

snr = 10.^(snr\_dB./10);

Pf = 0.01:0.01:1;

uc = 0.1;

%% BPSK Signal

L=1500;

data = round(rand(1,L)); % Data sequence

uni2bip=2\*data-1; % Convert unipolar to bipolar

T=1; % Bit duration

Eb=T/2; % This will result in unit amplitude waveforms

fc=3/T; % Carrier frequency

t=linspace(0,5,1500); % discrete time sequence between 0 and 5\*T (15000 samples)

K=length(t); % Number of samples

Nsb=K/length(data); % Number of samples per bit

dd=repmat(data',1,Nsb); % replicate each bit Nsb times

bb=repmat(uni2bip',1,Nsb); dw=dd'; % Transpose the rows and columns

dw=dw(:)';

%------ Convert dw to a column vector (colum by column) and convert to a row vector

bw=bb';

bw=bw(:)'; % Data sequence samples

w=sqrt(2\*Eb/T)\*cos(2\*pi\*fc\*t); % carrier waveform

bpsk\_w=bw.\*w; % modulated waveform

%% Probabilty of double threshold

u=1000;

Pfa = Pf;

for m=1:5

for i =1:length(Pf)

Dcv=0;

Dps0=0;

Dps1 = 1;

for j = 1:1000

% lambda(i)=gammaincinv(1-Pf(i),u)\*2; %theshold

% Pdth(i)=marcumq(sqrt(2\*snr),sqrt(lambda(i)),u);

%-----AWGN noise with mean 0 and variance -----%

Noise = randn(1,N);

vn=var(Noise);

%-----Real valued Gaussian Primary User Signal------%

Signal = sqrt(snr).\*bpsk\_w(1:200);

vs=var(Signal);

Recv\_Sig = Signal + Noise; % Received signal at SU 1

Energy = abs(Recv\_Sig).^2; % Energy of received signal over N samples

%------- Threshold-----------

Threshold\_0(i) = N\*vn + qfuncinv(Pf(i))\*sqrt(2\*N\*vn^2);

Threshold\_1(i) = (1-uc)\*Threshold\_0(i);

Threshold\_2(i) = (1+uc)\*Threshold\_0(i);

%------------------------------------

% Threshold = N\*vn + qfuncinv(Pf)\*sqrt(2\*N\*vn^2);

% Pdth = qfunc((Threshold -N\*(vn+vs))./(sqrt(2\*N\*(vn+vs)^2)));

%-----Computation of Test statistic for energy detection-----%

X(m) =sum(Energy);

%% ------------------Conventional---------------------

if X(m) >= Threshold\_0(i)

Dcv = Dcv +1;

end

%% Proposed scheme

if X(m) <= Threshold\_1(i)

%Pfa(i) = qfunc((Threshold\_0(i)-N\*vn)./(sqrt(2\*N\*vn^2)));

FC(m) = 0;

Dps0 = Dps0 +1 ;

elseif X(m) >= Threshold\_2(i)

FC(m) = 1;

%Pdsim(m,i) = qfunc((Threshold\_2(i) -N\*(vn+vs))./(sqrt(2\*N\*(vn+vs)^2)));

Dps1 = Dps1 + 1;

else

FC(m) = X(m);

end

end

Pdcv(m,i) = Dcv/j;

Pdps(m,i) = Dps1/j;

Pfps(m,i) = Dps0/j;

end

end

K=0;

FC\_E=0;

for i=1:5

if FC(i)==1 || FC(i)==0

K=K+1;

Lf(K)=FC(i);

else

FC\_E=FC\_E+FC(i);

end

end

W = 5-K;

if W~=0

X\_avg = FC\_E/W;

if X\_avg < Threshold\_0(K+1)

Lf(K+1) = 0;

else

Lf(K+1) = 1;

Pdcv(K+1,i) = qfunc((Threshold\_0(K+1) -N\*(vn+vs))./(sqrt(2\*N\*(vn+vs)^2)));

end

end

for i=1:100

PdcvOR(i)=1-prod(1-Pdcv(:,i));

PdpsOR(i)=1-prod(1-Pdps(:,i));

end

plot(Pf,PdcvOR,'b-o',Pf,PdpsOR,'g-\*');

grid on

axis([0.0001,1,0.0001,1]);

xlabel('Cooperative probability of False alarm (Qfa)');

ylabel('Cooperative Probability of Detection (Qd)');

legend('Conventional scheme','Proposed scheme');

### **3.4 Pe vs SNR && Qd vs SNR MATLAB Code:**

clc;

clear all;

format long

N = 200;

snr\_dB =-8; %dB

snr = 10.^(snr\_dB./10);

Pf = 0.01:0.01:1;

uc = 0.1;

%% BPSK Signal

L=1500;

data = round(rand(1,L)); % Data sequence

uni2bip=2\*data-1; % Convert unipolar to bipolar

T=1; % Bit duration

Eb=T/2; % This will result in unit amplitude waveforms

fc=3/T; % Carrier frequency

t=linspace(0,5,1500); % discrete time sequence between 0 and 5\*T (15000 samples)

K=length(t); % Number of samples

Nsb=K/length(data); % Number of samples per bit

dd=repmat(data',1,Nsb); % replicate each bit Nsb times

bb=repmat(uni2bip',1,Nsb); dw=dd'; % Transpose the rows and columns

dw=dw(:)';

%------ Convert dw to a column vector (colum by column) and convert to a row vector

bw=bb';

bw=bw(:)'; % Data sequence samples

w=sqrt(2\*Eb/T)\*cos(2\*pi\*fc\*t); % carrier waveform

bpsk\_w=bw.\*w; % modulated waveform

%% Probabilty of double threshold

u=1000;

Pfa = 0.1;

snr\_dB =-15:1:-5; %dB

snr = 10.^(snr\_dB./10);

for m=1:5

for i =1:length(snr)

Dcv=0;

Dps0=0;

Dps1 = 1;

for j = 1:1000

% lambda(i)=gammaincinv(1-Pf(i),u)\*2; %theshold

% Pdth(i)=marcumq(sqrt(2\*snr),sqrt(lambda(i)),u);

%-----AWGN noise with mean 0 and variance -----%

Noise = randn(1,N);

vn=var(Noise);

%-----Real valued Gaussian Primary User Signal------%

Signal = sqrt(snr(i)).\*bpsk\_w(1:200);

vs=var(Signal);

Recv\_Sig = Signal + Noise; % Received signal at SU 1

Energy = abs(Recv\_Sig).^2; % Energy of received signal over N samples

%------- Threshold-----------

Threshold\_0(i) = N\*vn + qfuncinv(Pfa)\*sqrt(2\*N\*vn^2);

Threshold\_1(i) = (1-uc)\*Threshold\_0(i);

Threshold\_2(i) = (1+uc)\*Threshold\_0(i);

%-----Computation of Test statistic for energy detection-----%

X(m) =sum(Energy);

%% ------------------Conventional---------------------

if X(m) >= Threshold\_0(i)

Dcv = Dcv +1;

end

%% Proposed scheme

if X(m) <= Threshold\_1(i)

FC(m) = 0;

Dps0 = Dps0 +1 ;

elseif X(m) >= Threshold\_2(i)

FC(m) = 1;

Dps1 = Dps1 + 1;

else

FC(m) = X(m);

end

end

Pdcv(m,i) = Dcv/j;

Pdps(m,i) = Dps1/j;

Pfps(m,i) = Dps0/j;

end

end

K=0;

FC\_E=0;

for i=1:5

if FC(i)==1 || FC(i)==0

K=K+1;

Lf(K)=FC(i);

else

FC\_E=FC\_E+FC(i);

end

end

W = 5-K;

if W~=0

X\_avg = FC\_E/W;

if X\_avg < Threshold\_0(K+1)

Lf(K+1) = 0;

else

Lf(K+1) = 1;

Pdcv(K+1,i) = qfunc((Threshold\_0(K+1) -N\*(vn+vs))./(sqrt(2\*N\*(vn+vs)^2)));

end

end

for i=1:length(snr)

PdcvOR(i)=1-prod(1-Pdcv(:,i));

PdpsOR(i)=1-prod(1-Pdps(:,i));

end

plot(snr\_dB,PdcvOR,'b-o',snr\_dB,PdpsOR,'g-\*');

grid on

axis([-15.001,-5,0.0001,1]);

xlabel('SNR(in dB)');

ylabel('Cooperative Probability of Detection (Qd)');

legend('Conventional scheme','Proposed scheme');

figure

for i=1:length(snr\_dB)

Pecv(i)=Pfa+1-PdcvOR(i);

Peps(i)=Pfa+1-PdpsOR(i);

end

plot(snr\_dB,Pecv,'b-o',snr\_dB,Peps,'g-\*');

grid on

xlabel('SNR(dB)');

axis([-15.001,-5,0.0001,1])

ylabel('Probability of Error (Pe)');

legend('Conventional scheme','Proposed scheme');