Application of Software Defined Radio in improving SNR by Empirical Mode Decomposition

By

Sapan H Mankad

08MCES57

Department of Computer Science and Engineering

Institute of Technology, Nirma University

Ahmedabad

May 2011

Application of Software Defined Radio in improving SNR by Empirical Mode Decomposition

Major Project

Submitted in partial fulfillment of the requirements
for the degree of
Master of Technology in Computer Science and Engineering

By
Sapan H Mankad
08MCES57

Guided By
Dr S N Pradhan

Department of Computer Science and Engineering

Institute of Technology, Nirma University

Ahmedabad

May 2011

Declaration

This is to certify that

i) The thesis comprises my original work towards the degree of Master of Technology in Computer Science and Engineering at Nirma University and has not been submitted elsewhere for a degree.

ii) Due acknowledgement has been made in the text to all other material used.

Sapan H Mankad

Certificate

This is to certify that the Major Project Part-II entitled ”Application of Software Defined Radio in improving SNR by Empirical Mode Decomposition” submitted by Sapan H Mankad (08MCES57), towards the partial fulfillment of the requirements for the degree of Master of Technology in Computer Science and Engineering of Nirma University, Ahmedabad is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this Major Project Part-II, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Dr S N Pradhan
Guide and Professor,
Department Computer Science & Engineering,
Institute of Technology,
Nirma University, Ahmedabad

Prof D J Patel
Professor and Head,
Department Computer Science & Engineering,
Institute of Technology,
Nirma University, Ahmedabad

Dr K Kotecha
Director,
Institute of Technology,
Nirma University,
Ahmedabad

Abstract

Software Defined Radios(SDR) are aimed at reducing the efforts required specifically in Wireless Communication. Many hardware devices are currently being used for communicating via radio waves.The function of SDR is to replace possibly all the hardware stuff by software which results in great flexibility and portability.This concept has opened new windows to the world of digital communication.There has been in existence many flavors of such radios recently. This thesis has concentrated on the open source GNU Radio during the work.The GNU Radio Project is related to testing various capabilities of software radios and serves as a testbed for such innovations.This thesis deals with utilizing the capabilities of software radios to improve the quality of the incoming signal.Our inclination was towards optimizing the received signal by menas of reducing noise and thus enhancing the overall communication.We present and propose the use of Empirical Mode Decomposition method.This method helps separating out the noise components from the signal so as to improvise.

Acknowledgements

I would like to thank all the faculty members of the Computer Science and Engineering department for their assistance with this thesis, particularly my thesis advisor **Dr S N Pradhan** for introducing me to the very interesting concept of Software Defined Radios, for suggesting this topic to me and for all the helpful suggestions, and most importantly for his motivating enthusiasm forever. While giving his students a large degree of independence and flexibility to manage their time and projects, he is always available and willing to help in any way possible.

I am also grateful to **Prof.Manish Chaturvedi**, for encouraging me to pursue my research topic and for his guidance over the course of my time at the University. I also express my profound gratitude to **Prof.Zunnun Narmawala** and **Prof. Nirav Patel** for their timely suggestions,continuous support and help to solve some of the doubts.

My sincere thanks are due to the members of the GNU Radio project, for producing this excellent software that made my research possible and and all those who have responded to my questions and become helping hands via the GNU Radio mailing list during any technical difficulty.I also acknowledge my gratefulness to the contribution of journals,papers,organizations and books which have been very useful learning materials during my work.

Finally, there were of course those who did not directly contribute to this thesis, but who supported me in various ways during the writing of my thesis and the duration of my study. I would like to dedicate this thesis to my family. I have no words to convey my thoughts to appreciate what they have done for me. Thank you for being so supportive and helping me throughout my academic career.

- SAPAN H MANKAD
08MCES57

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Mobile communications have expanded the horizons of signal processing in the modern era of communication technologies.Several aspects of digital and analog signal processing affect the performance of communication.To achieve benefits,the minimum expected performance has to be met by concerned organization or tele-communication companies or their signal processing logic.Nowadays,when the planet is connected all over,*mobility* has also become common to all.The issues related to mobility can weaken the performance of the signals being transmitted.In addition,in the technologically rapidly emerging world,new applications arrive at quick rate which everyone wants to avail.*Reconfigurability* is another factor associated in this concern.The target is that the incoming signal reaches with minimum possible attenuation to the destination.Mechanisms to smoothen the received signal have to be *portable* or ready to use.This can be beneficial not only to improve the performance but also achieve better signal reception quality.

With the above discussed parameters in consideration,I have worked on my thesis during this period.My task was to explore the new possibilities through which QoS parameters for the incoming signal can be optimized,especially in case of mobile communication signals.In general terms,signal processing is done through hardware based radio consisting of the components as illustrated in Figure 1.1.The use of hardware circuitry limits the researchers to make any dynamic change frequently.This difficulty

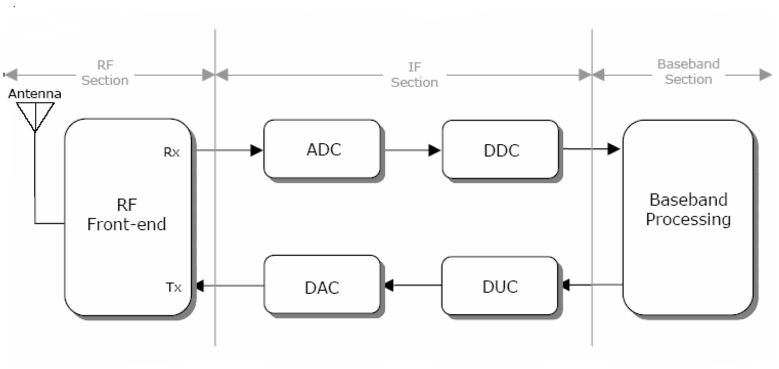


Figure 1.1: Block Diagram for Hardware Defined Radio

led to search some alternative way to accomplish the same task more fairly.This is how Software Defined Radio(SDR) came into existence.It brings the capabilities of radio functionalities with signal processing functionalities to achieve reconfigurability and portability.More about SDR has been discussed in Chapter.

My target was to make it possible to utilize the capabilities of SDR to reduce the hardware based tasks and maximize software based computations so as to achieve reconfigurability.SDR,thus in real terms, enables *R*econfigurable *A*daptive *D*ynamic *I*nput *O*utput.

1.1 Objective

The objective of the work was to improve the incoming signal quality by means of reducing noise portion from it using SDR.I used Empirical Mode Decomposition method as a base for this task.

1.2 Scope of the Work

The scope of this work is to optimize a QoS parameter during communication.The target was to achieve better performance for varying signal to noise ratios.I focused on BER and SNR as major parameters to deal with during my experiment.

1.3 Motivation of the Work

Many researchers have thoroughly gone through the use of GNU Radio,an open source SDR and contributed their findings on applications such as wireless protocol testing or implementing various receivers.But a little research has been done in the area of writing a custom signal processing application using this tool.The challenge to me was to find a suitable way for noise reduction from an incoming signal,thus reducing its BER.I found that till date,there haven't been any inbuilt signal processing block in GNU Radio to support this task except usual filtering blocks(which in turn,requires input parameters such as frequency or the pattern structure of the noise,which may not always be possible to estimate).Especially in wireless domain,one cannot predict the noise pattern in advance.My work was centered around exploring or establishing some method that could help solve this problem.I found Empirical Mode Decomposition[6] which can decompose the input signal into monotonic signals,which may later be useful to identify the noisy components.Section describes this method in detail.

1.4 Thesis Organization

The rest of the thesis is organized as follows.

Chapter 2, *Literature Survey*, describes the concept of software defined radios,the reasons underlying their need to replace hardware and some of their applications.This chapter further discusses the methodology behind Empirical Mode Decomposition technique and its theoretical background.

Chapter 3, *GNU Radio*,includes the study of the open source software defined radio,its architecture.This chapter also deals with the procedure to install GNU Radio,introduces a GUI tool,referred to as GNU Radio Companion.At the end,we discuss the main theme of the thesis i.e;the procedure to write a custom block in GNU Radio.

Chapter 4, *Proposed Scheme*, introduces our scheme on using Empirical Mode Decomposition to improve the bit error rate.

In **Chapter 5**, *Results and Discussion*,the simulation scenario is depicted and the results are analyzed in terms of parameters such as Bit Error Rate(BER) and Signal to Noise Ratio(SNR) for speech signal and GSM modulated signal.

Finally, in **Chapter 6** *Conclusion and Future Scope*,concluding remarks and future scope is presented.

Chapter 2

Literature Survey

2.1 Software Defined Radio

2.1.1 Foundation of SDR

Software Defined Radio - this term was coined by Dr.Joseph Mitola in 1991[3].As described in [4],"a basic SDR system may consist of a personal computer equipped with a sound card, or other analog-to-digital converter, preceded by some form of RF front end". Significant amounts of signal processing are handed over to the general-purpose processor, rather than being done in special-purpose hardware. Such a design produces a radio that can receive and transmit widely different radio protocols (sometimes referred to as a waveforms) based solely on the software used. SDR allows a single wireless device to support a wide range of capabilities previously available only by integrating multiple radio components.Compared to traditional hardware-oriented approaches, such as DSP and FPGA-based solutions, a true SDR base station is highly modular and enables a high degree of software portability and reuse, minimizing the amount of code that has to be re-written to keep pace with advances in the underlying technology.

Software Defined Radio (SDR) until now has been seen as a military technology with a limited market for commercial applications. The commercial use of SDR has been restricted to providing partial software upgradeability within a given family of wireless standards. This has been due to technological bottlenecks at the RF front end and its inability to be reconfigurable. However, with recent innovations in enabling wideband RF front ends and soft transceivers,SDR can move beyond 'partial reconfigurability' to 'multiprotocol multiband reconfigurability'[10].An SDR refers to a class of radios,the capabilities of which are not simply provided by software but utilize an infrastructure that supports interchangeable components as well as functionalities.

2.1.2 Need for SDR

Due to the operating trend and capabilities of general purpose processors(GPP), the processing power of DSP and FPGA,more of radio signaling is performed by software. This is the main reason behind the fact that radio systems have become more flexible as more capabilities are provided via software. Let us discuss some of the reasons behind increasing trend of software defined radios.One reason is that legacy based hardware radios could not be reconfigured without physical modifications or redesign.Another issue is that the radio system could not be managed,configured or controlled using a consistent set of interfaces and protocols.Further,no interoperability is supported due to different manufacturers.

In order to address these problems,US government initiated a series of programmes leading towards the specification of a common software infrastructure for software defined radios.It started into mid 1990s and evolved into the Software Communications Architecture(SCA).It is the first specification that represents the combined contributions of many of the key radio system manufacturers from US.

The fundamental objective of SCA is to provide a common software infrastructure for managing radio systems.The software is loaded and controlled through proprietary mechanisms and each radio manufacturer employs a unique infrastructure or architecture.The SCA specification describes a collection of components,their configuration

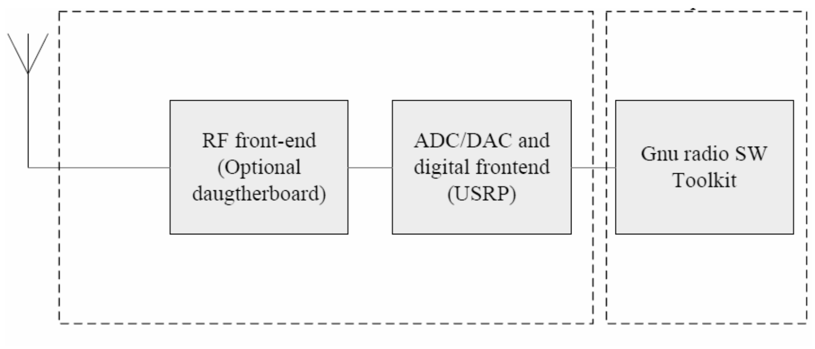

RF front-end (Optional daugtherboard)
ADC/DAC and digital frontend (USRP)
Gnu radio SW Toolkit

Figure 2.1: SDR Architecture

and the assembly into a functional waveform application on a radio system.All these form an infrastructure for defining and constructing an SDR system.JPEO JTRS (Joint Tactical Radio System) standards website [1] provides services for downloading the SCA,APIs and other documentation.

2.1.3 SDR Applications

Software radio has found uses in amateur radio, academia, industry, government, and military applications. In communications research laboratories, software radio is being used for testing and evaluation of multihop and mobile multimedia broadcasting transmissions, wireless network experimentation and cross-layer prototyping, and so-called cognitive radio research and development.

2.1.4 SDR Variants

There are many variants of SDR developed by some organizations as well as researchers.Some of them include HPSDR(High Performance SDR), Vanu Radio, Flex Radio,SORA, OSSIE, GNU Radio etc.It was found that GNU Radio provides a powerful toolbox for signal processing that can be easily upgraded.Chapter 3 discusses about the GNU Radio in detail.

2.2 Empirical Mode Decomposition

2.2.1 Mathematical Concept

A complex signal should consist of some simple signals, each of which involves only one oscillatory mode at any time instance.

The Empirical Mode Decomposition (EMD) is an iterative process which decomposes real signals $f(t)$ into simpler signals (modes)

$$f(t) = \sum_{j=1}^{M} \phi_j(t) \tag{2.1}$$

Each monocomponent signal ϕ_j should be representable in the form

$$\phi(t) = r(t) sin\theta(t) \tag{2.2}$$

These monocomponent signals ϕ, called Intrinsic Mode Functions or IMFs, are produced by the Empirical Mode Decomposition.

EMD decomposes the signal according to the signal itself and the number of the IMFs is finite. Furthermore, IMFs reflect the intrinsic and reality information of the analyzed signal. Therefore, EMD method is a self-adaptive signal-processing method that is suitable for the analysis of non-linear and non-stationary process[9].

The original objective of EMD was to identify the intrinsic oscillatory modes in each time location from a signal, one by one. With EMD, any complicated signal can be decomposed into a finite number of simple signals, each of which includes only one oscillatory mode in any time location. These extracted simple signals actually serve as approximations of so-called mono-component signals. However, it is difficult to tell what is an intrinsic oscillatory mode of a signal in a time location.

2.2.2 Definition of Intrinsic Mode Function

Intrinsic Mode Function A function $\lambda(t)$ is defined to be an Intrinsic Mode Function[15], or more briefly an IMF, of a real variable t, if it satisfies two characteristic properties:

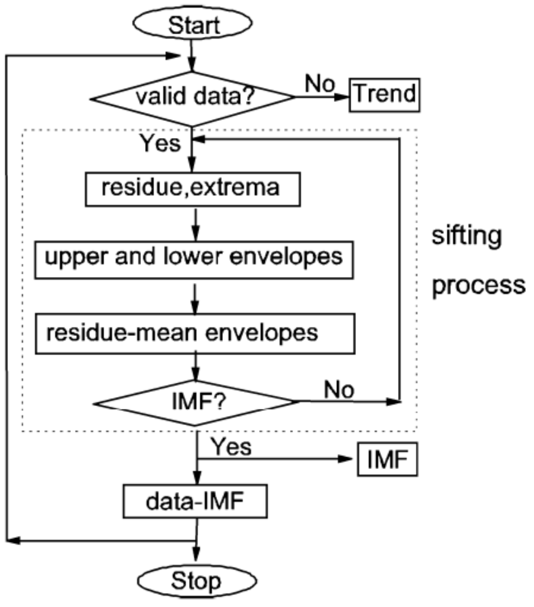


Figure 2.2: The flow chart of the Sifting Process[7]

1. λ has exactly one zero between any two consecutive local extrema.
2. λ has zero local mean.

A function which is required to only satisfy condition (a) will be called a weak-IMF.

2.2.3 Sifting Process

EMD as originally proposed is implemented through a sifting process,which is described as follows:

1. For any given data, $x(t)$, we identify all the local extrema.
2. Separately connect all the maxima and minima with natural cubic spline lines to form the upper, $u(t)$, and lower, $l(t)$,envelopes.
3. Find the mean of the envelopes as $m(t) = [u(t) + l(t)]/2$.

4. Take the difference between the data and the mean as the proto-IMF, $h(t) = x(t) - m(t)$.

5. Check the proto-IMF against the definition of IMF and the stoppage criterion to determine if it is an IMF.

6. If the proto-IMF does not satisfy the definition, repeat step a to e on $h(t)$ as many time as needed till it satisfies the definition.

7. If the proto-IMF does satisfy the definition, assign the proto-IMF as an IMF component,$c(t)$.

8. Repeat the operation step 1 to 7 on the residue, $r(t) = x(t) - c(t)$, as the data.

9. The operation ends when the residue contains no more than one extremum.

The flowchart for calculating IMFs from a complex signal is described in Figure 2.2.

The results of [17] show that the IMF plots reveal that when noise is added to a clean speech,the first few IMFs contain most of noise energy and some of the speech. Our goal is to distinguish which IMF contain the speech or noise.This decomposition pushes a significant amount of the speech energy to latter IMFs along with some residual noise.The reconstruction process is given in Equation 2.3,which involves combining the n IMFs and the residual r[n].

$$x[n] = \sum_{i=1}^{n} IMF[n] + r[n] \tag{2.3}$$

The original EMD is obtained through an algorithm called the sifting process. The local maxima and minima in the process are respectively connected through cubic splines to form the so-called upper and lower envelopes. The average of the two envelopes is then subtracted from the original data. EMD is obtained after applying this process repeatedly. The sifting algorithm is highly adaptive; it is also unstable. A small change in data can often lead to different EMD. Many fundamental mathematical issues such as the convergence of the sifting algorithm, the orthogonality of IMFs and others have never been established. The difficulty is partly due to the highly adaptive nature of the sifting algorithm as well as the ad hoc nature of using cubic splines. In [14], the cubic splines were replaced by B-splines, which gives an alternative way for EMD. But again this modification does not resolve those mathematical issues.The convergence problem has been addressed in [11] using iterating filters,but it provides similar results.[9] show that the IMFs defined by their energy difference tracking method meet the orthogonality condition and reflect the intrinsic and reality information of the analysed signal.

The authors in [13] discusses a very good comparison of different assessing alternatives for the sifting process and introduces the use of rational splines which results into tradeoffs relative to the original cubic spline method.It is succeeded to reduce the over- and undershooting problems,but at the expense of more IMFs and more sifting.

One of the main assets of the EMD is its sparseness: an arbitrary signal is decomposed in fewer components than with classical Fourier or even wavelet analysis. Another important asset is the completeness: by design, the algorithm guarantees a lossless decomposition. However, one of the main drawbacks is the non uniqueness of the final decomposition. It is indeed strongly dependent on the different algorithm parameters and choices, like the sifting ending criterion, the boundaries ending technique (signal continuation), the interpolation method, etc.

2.3 Summary

The Empirical mode decomposition (EMD) method is an algorithm for the analysis of multicomponent signals that works by breaking the signal into a number of amplitude and frequency modulated (AM/FM) zero mean signals, termed intrinsic mode functions (IMFs). Thus,above discussion yields that no method has any proven convergence criterion or a specific recommendation for which spline interpolation to use.Hence,we go for the further process unchanged,with the naive EMD,using cubic interpolation.

Chapter 3

GNU Radio

3.1 Overview

Eric Blossom is the founder and maintainer of GNU Radio Project[2] worldwide.Many researchers are involved in this project.Matt Ettus(http://www.ettus.org)has developed a low cost hardware,USRP, to work with GNU Radio that enables it to work with over the air signals.Josh Blum is another person who has developed a great user interface(GNU Radio Companion) to simplify GNU Radio programming.GNU Radio is an open source software framework for writing SDR applications under Linux, Unix, Windows or Mac OS X. It provides a set of base classes from which custom applications can be derived.In addition,GNU Radio provides a host of signal processing, hardware interface, graphical user interface, and utility libraries.Figure 2.1 depicts the architecture of GNU software radio.

3.2 GNU Radio Framework

The GNU Radio framework is built around the graph design pattern.Using GNU Radio, a radio can be built by creating a graph where the vertices are signal processing blocks and the edges represent the data flow between them.Figure 3.1 describes the components of the GNU Radio. The signal processing blocks are implemented in

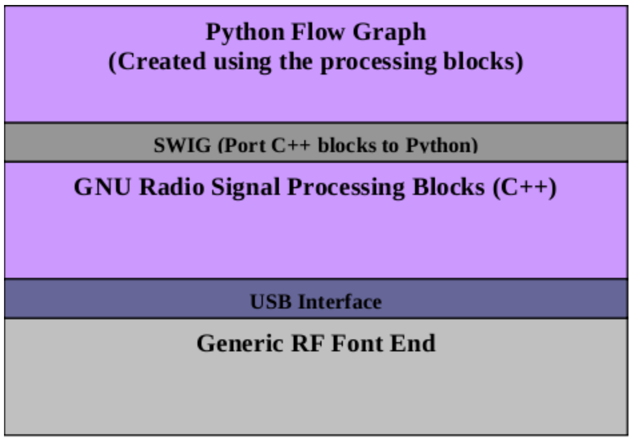

Python Flow Graph
(Created using the processing blocks)
SWIG (Port C++ blocks to Python)
GNU Radio Signal Processing Blocks (C++)
USB Interface
Generic RF Font End

Figure 3.1: GNU Radio Components

C++ and the graphs are constructed and run in Python.

SWIG works as glue between C++ classes and Python language. SWIG is a Linux package that converts the C++ classes into Python compatible classes. This way, GNU Radio framework is able to harness both languages. While C++ provides compact code for signal processing block, Python is preferred because of its flexibility and ease to program.

Conceptually, a signal processing block processes an infinite stream of data flowing from its input ports to its output ports. A block's attributes include the number of input and output ports it has as well as the type of data that flows through each. Some blocks have only output ports or input ports. Input and output ports serve as data sources and sinks in the graph.Thus,a radio is built by creating a graph where the vertices are signal processing blocks and the edges represent the data flow between them.

GNU radio software modules support various modulation schemes,error correction codes and signal processing capabilities such as filters,FFTs,equalizers and timing recovery.

3.3 GNU Radio Installation

GNU Radio source is available from its website for typically Linux users.There are many alternatives to install this package.

GNU Radio Tarball After downloading the tar file from the website, some necessary steps are required to execute.This is described in Appendix.

Synaptic Package Manager This facility works as a wizard for Ubuntu users in which the online user needs to just select the package to install and it gets installed automatically.I used this option to install the package.

Command Line Installation By executing following command at the terminal,the task can be done.

```
sudo apt-get install gnuradio gnuradio-companion
```

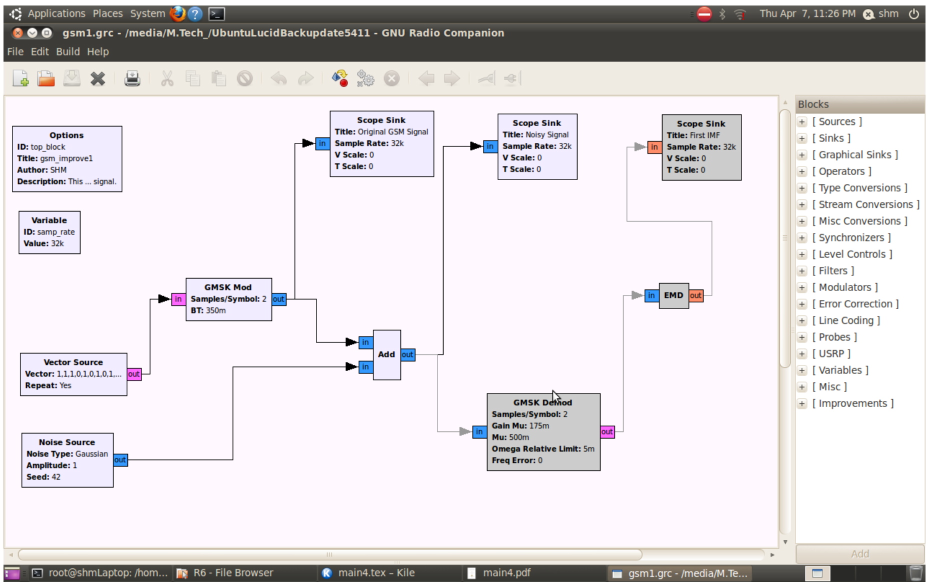
I used Ubuntu 10.04 Lucid for my installation.

Prior to this installation,it is mandatory to install some of the recommended prerequisite packages.A very useful document about the same is provided in [12].

Initially,I faced many difficulties installing GNU Radio on my machine.I was not able to get output in terms of video.This was due to some problem in the package wx_Python.Later,I could sort out the problem.At one stage,I was not able to open GRC,even after successfully installing it.Finally,I solved that problem also by reinstalling the entire package.Thus,during my thesis,I have reinstalled this package about six to seven times and changed my OS from Ubuntu Karmic(9.10) to Lucid(10.04) due to one or another reason many times.At present,everything works well on my Lucid.

3.4 GNU Radio Companion

GNU Radio Companion(GRC) is a handy application developed by Josh Blum to ease the programming.In this formation,signal processing blocks are written in C++ and using the SWIG(Simplified Wrapper and interface Generator) tool,mapped into Python.GRC follows a graph structure,in which blocks are the vertices of the graph and links connecting these blocks are the edges of the graph.



Options
ID: top_block
Title: gsm_improve1
Author: SHM
Description: This ... signal.

Variable
ID: samp_rate
Value: 32k

Vector Source
Vector: 1,1,1,0,1,0,1,0,1,...
Repeat: Yes

Noise Source
Noise Type: Gaussian
Amplitude: 1
Seed: 42

GMSK Mod
Samples/Symbol: 2
BT: 350m

Add

Scope Sink
Title: Original GSM Signal
Sample Rate: 32k
V Scale: 0
T Scale: 0

Scope Sink
Title: Noisy Signal
Sample Rate: 32k
V Scale: 0
T Scale: 0

GMSK Demod
Samples/Symbol: 2
Gain Mu: 175m
Mu: 500m
Omega Relative Limit: 5m
Freq Error: 0

EMD

Scope Sink
Title: First IMF
Sample Rate: 32k
V Scale: 0
T Scale: 0

Figure 3.2: Flowgraph for Simulated Environment

Figure 3.2 shows the GRC flowgraph for simulation scenario for our model.

3.4.1 Testing Applications

As a *Hello World* program,to test the successful installation of GNU Radio and GRC,we executed many flowgraphs such as Dialtone program,AM receiver,FM receiver,Filtering a signal and we found this tool very useful.If USRP is available,live captures can be possible over the air and our PC may turn into AM/FM radio.For our sake,we used recorded samples available from [8].

3.5 How to write a customized block?

Our target was to add a custom block that implements Empirical Mode Decomposition.Eric Blossom has kept a very resourceful document [5] at the GNU Radio website.Alongwith that,other useful tutorials from this website helped proceeding with the goal.

PURPOSE	**FILE TO BE CREATED**
Specify parameters for the blocks	gr_my_emd.xml
Specify class and variable declarations	gr_my_emd.h
Specify class definition and actual code	gr_my_emd.cc
Automatic Conversion from C++ code into Python code	swig.i

Table I: Necessary files for adding custom block in GNU Radio

Table I states that some files are to be created in order to successfully implement the custom block and use it through GRC.

3.6 Configuring new block

Configuring new block in GNU Radio involves several files to be created, needs to place them at proper place and execute certain commands.This section describes the required procedure for this task.

As mentioned in the GNU Radio website,some folders in the root directory have to be used,namely as follows:

apps Contains test applications and examples.

config Contains different configuration files which must not be modified.

lib Contains source files (.cc and .h) for the developed blocks.

swig As GNU Radio combines C++ programmed blocks with Python flowgraphs, Swig tool is employed to automatically generate python interfaces for C++ blocks. .i files required by this tool are stored in this folder.

python Contains python scripts.

grc Contains the .xml files needed by the grc tool.

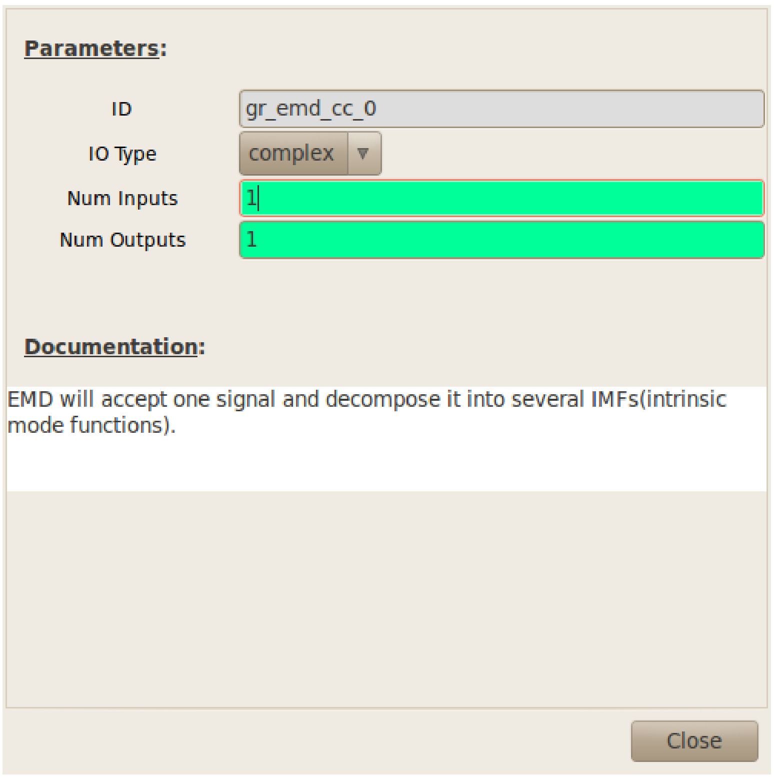
Each folder contains a Makefile.am file because autotools (autoconf, automake and libtool) are employed in order to automatically compile and link.

3.6.1 Specify schema of the block

The block parameters can be given by generating a simple xml schema file,which indicates the number and types of input/output to the block.In addition,it also takes care of dynamic setting of any useful parameter.We created the file $gr_my_emd.xml$ which is shown in the following contents:

Figure 3.3 shows the basic parameters involved with our customized block in GNU Radio Companion.

```
 <?xml version="1.0"?>
<!-- #### Empirical Mode Decomposition ### -->
<block>
    <name>EMD</name>
    <key>gr_emd_cc</key>
    <category>Improvements</category>
    <import>from gnuradio import gr</import>
    <make>gr.emd_v$(type.fcn)($vlen)</make>
    <param>
        <name>IO Type</name>
        <key>type</key>
        <type>enum</type>
```


Parameters:

ID: gr_emd_cc_0
IO Type: complex
Num Inputs: 1
Num Outputs: 1

Documentation:

EMD will accept one signal and decompose it into several IMFs(intrinsic mode functions).

Figure 3.3: Parameters for our custom block

```
        <option>
            <name>complex</name>
            <key>complex</key>
            <opt>fcn:cc</opt>
        </option>
        <option>
            <name>Float</name>
            <key>float</key>
            <opt>fcn:ff</opt>
        </option>
    </param>
    <param>
```

```
        <name>Num Inputs</name>
        <key>num_inputs</key>
        <value>1</value>
        <type>int</type>
    </param>
    <param>
        <name>Num Outputs</name>
        <key>num_outputs</key>
        <value>1</value>
        <type>int</type>
    </param>
    <check>0 &lt; $num_outputs</check>
    <sink>
        <name>in</name>
        <type>$type</type>
        <nports>$num_inputs</nports>
    </sink>
    <source>
        <name>out</name>
        <type>float</type>
        <nports>$num_outputs</nports>
    </source>
    <doc>
EMD will accept one signal and decompose it into several
IMFs(intrinsic mode functions).

    </doc>
</block>
```

After saving this xml file in the GNU Radio tree at the appropriate place,the desired block gets added into GRC blocks under a new category *Improvement* in this case.

3.6.2 Setting up the block

In order to compile and deploy the custom module,it is necessary to execute the following commands in the exact order.

```
#sudo ./bootstrap

#sudo ./configure

#cd swig

#sudo make generate-makefile-swig (this is done for regenerating the
Makefile.swig.gen file in the swig folder)

#cd ..

#sudo make

#sudo make install

#sudo ldconfig (only the first time the module is installed)
```

Once these commands are successfully executed,our block is ready to be used from gnuradio companion interface.

Chapter 4

Proposed Scheme

Improvement over noise can be done by applying different kinds of filters on the signal;but in this case,we have to have prior knowledge about the structure of the noise which is hardly possible in case of wireless networks.Hence,it was essential to design or explore a new method to accomplish this task to achieve better quality signal at the receiver side. After some research,we found that Empirical Mode Decomposition

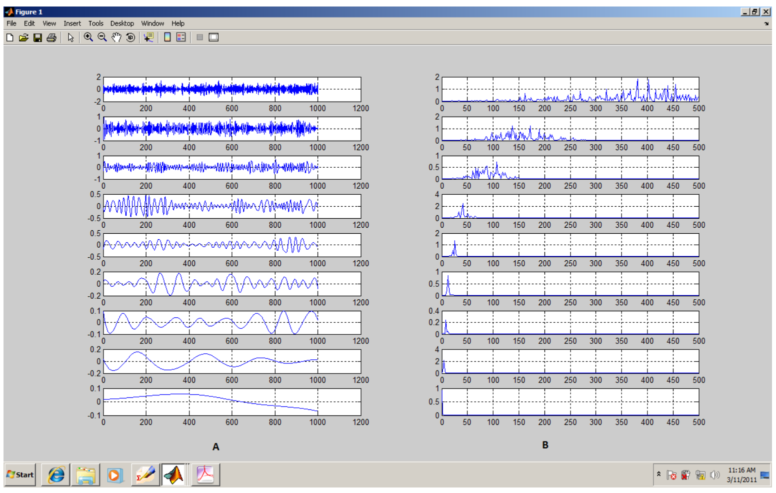


Figure 4.1: IMFs and their Power Spectrum

[6],is a method that works on nonstationary signals and can be useful to find estimates of noisy portion from the signal.Section 2.2 discussed about this technique and its

variants in detail.

[16] describe in their work the use of Empirical Mode Decomposition for denoising signals in an efficient manner.They suggest that it should be possible to separate out the noise portion from the incoming signal.

Figure 4.1 shows the intrinsic mode functions for the speech signal and their corresponding frequency distributions.The observation yields that initial IMFs contain most of the noise components,while the later IMFs are more inclined towards the information part.

We also proceed with the same theme.Our task was to observe the noise component allocation in the signal under operation and remove the most dominant noisy IMF from those extracted and sum up all the remaining IMFs.

Chapter 5

Results and Discussion

5.1 Experimental Setup

A typical communication link includes, at a minimum, three key elements: a transmitter, a communication medium (or channel), and a receiver. The ability to simulate all these functions is required to successfully model any end-to-end communication system.As our goal was oriented towards improvement of the signal using Empirical Mode Decomposition,we used Matlab at the initial stage to test EMD and its effect over the noisy signals.Table I depicts the typical simulation scenario for our experiment.

1. Generate a random bit stream.
2. Modulate it using GMSK.The resultant signal is GSM signal x.
3. Simulate noisy channel by adding different types of noise. The noisy signal is y.
4. The receiver receives signal and after demodulation,it is denoted as z.
5. If EMD is applied on y prior to demodulation,it becomes z_{emd}.
6. Compare z and z_{emd}.

Table I: Simulation Scenario

5.1.1 Simulation on Speech Signal

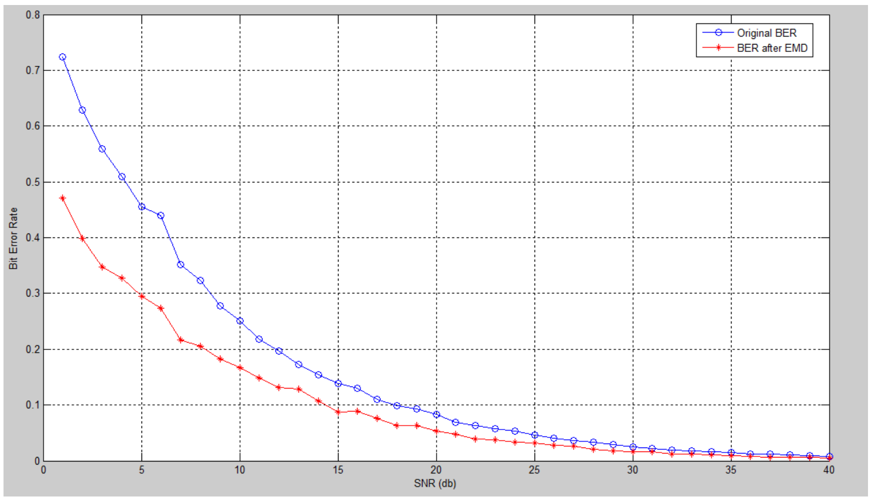


Figure 5.1: Effect of EMD over Speech Signal

The simulation result of [17] shows that for low E_b/N_o,EMD method improves BER performance by approximately 3dB gain.These BER improvements can help solving the problem related to call drop outs and improves overall QoS.

In the beginning,we used a sample speech segment to work with.The signal was passed through AWGN channel with varying SNR values.The EMD processed signal and unprocessed signal show significant difference.This is depicted using Figure 5.1.

5.1.2 Simulating GSM Signal

At present,GSM and CDMA are maximally used mobile communication signals.Our approach was to explore the effect of different types of noise and EMD,as denoising technique and embedding these functionalities into software defined radio.

At first,we constructed GSM signal by passing a random binary pattern through GMSK modulator,provided inside GNU Radio.This signal was corrupted with three different kinds of noise:gaussian noise,uniform noise and random noise.After applying EMD on this noisy signal,we compared the corresponding outputs produced after the experiment.Table II discusses the outcomes of this work.

TYPE OF NOISE	TIME(sec)	BER	BER with EMD
Gaussian	33.5	0.908	0.83
Uniform	19.5	0.914	0.869
Random	17	0.873	0.883

Table II: Comparison of BER with and without EMD removing only the first IMF

TYPE OF NOISE	TIME(sec)	BER	BER with EMD
Gaussian	33.5	0.91	0.76
Uniform	19.2	0.914	0.734
Random	16.7	0.873	0.76

Table III: Comparison of BER with and without EMD removing first two IMFs

This simulation was handled for 2000 samples of GMSK signal and the results show that EMD performs better in presence of Gaussian Noise and it does not have any improvement during random noise.For GSM signals,it shows that for increasing SNR values,the BER remains the constant.

The underlying observation should indicate that EMD should be avoided while random noise is active.We conducted another experiment with a little change in the scenario.We extracted first two IMFs from the noisy signal and deducted them from it.The results after this modification are depicted in Table III.

TYPE OF NOISE	TIME(sec)	BER	BER with EMD
Gaussian	33.8	0.90	0.76
Uniform	19.4	0.92	0.74
Random	16.8	0.876	0.76

Table IV: Comparison with Nearest Neighbor Spline

The result after the modification shows that EMD has the same effect over Gaussian and Uniform noise,whereas there is a significant improvement over random noise in this case.This is because the most of the noise energy is concentrated in the initial

IMFs.This may vary for different signals,but ultimately it is contained within the first three IMFs.

TYPE OF NOISE	TIME(sec)	BER	BER with EMD
Gaussian	33.8	0.90	0.76
Uniform	19.2	0.92	0.74
Random	16.7	0.875	0.76

Table V: Comparison with Linear Spline

The naive algorithm for EMD uses cubic natural spline.We use results using nearest neighbor in TableIV and linear spline interpolation in TableV to check the effect of EMD.This shows that cubic interpolation is the most suitable approach for all the cases.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

Empirical Mode Decomposition provides better results for signals having low signal to noise ratios.In addition,it works well when we remove the first two IMFs from the signal under process.Results show good improvements for different kinds of noise.Simulation of GSM signal for the experiment provides a good insight that wireless communication can be made noise prone at some extent using method such as discussed in this work.This is very useful and can be extended to achieve improvements in live captured signals by the virtue of GNU Radio.

6.2 Future Scope

This work can be extended by implementing variants of EMD technique and providing more parameters to choose while performing the denoising using EMD inside GNU Radio.Implementation can be more realistic if any hardware such as USRP is attached to GNU Radio so as to directly receive the OTA(over the air) files and process them directly.

References

[1] Offficial jtrs website. http://jpeojtrs.mil/.

[2] Official gnu radio website. http://www.gnuradio.org.

[3] Official website of joseph mitola. http://web.it.kth.se/ maguire/jmitola/.

[4] John Bard and Vincent J. Kovarik. *Software Defined Radio: The Software Communications Architecture*. John Willey and Sons.

[5] Eric Blossom. How to write a signal processing block.

[6] N.Huang et al. The empirical mode decomposition and the hilbert spectrum for nonlinear nonsta- tionary time series analysis. Proceedings of Royal Society of London, 1998.

[7] XIAN-YAO CHEN GANG WANG. and FANG-LI QIAO. On intrinsic mode function. volume 2, pages 277–293. World Scientific Publishing Company, 2010.

[8] Michael Grey. http://www.kd7lmo.net.

[9] Cheng Junsheng, Yu Dejie, and Yang Yu. Research on the intrinsic mode function (imf) criterion in emd method. volume 20, pages 817–824. Elseveir, September 2005.

[10] Aditya Kaul. Software defined radio: The transition from defense to commercial markets. SDR 07 Technical Conference and Product Exposition, 2007.

[11] YANG WANG LUAN LIN and HAOMIN ZHOU. Iterative filtering as an alternative algorithm for empirical mode decomposition.

[12] Naveen Manicka. Gnu radio testbed. Master's thesis, University of Delaware, 2007.

[13] G.G.S. Pegram Peel, M.C. and T.A. McMahon. Empirical mode decomposition: Improvement and application.

[14] S. Riemenschneider Q. Chen, N. Huang and Y. Xu. B-spline approach for empirical mode decom- position.

[15] Robert C Sharpley and Vesselin Vatchev. *Analysis of the Intrinsic Mode Functions*. Industrial Mathematical Institute, Universitu of South Carolina.

[16] Stephen McLaughlin Yannis Kopsinis. Empirical mode decomposition based denoising techniques.

[17] Z.Guo, Z.Xu, F.Wang, and B.Huang. Empirical mode decomposition for ber improvement in cellular network. volume 9(1), pages 146–151. Asian Network for Scientific Information, 2010.