CHAPTER – 1

INTRODUCTION

1.1 Introduction

The electrical activity of the heart can be recorded by the process of Electrocardiography in which electrodes are placed on the skin and measure the potential difference between the points of measurement, which arise due to the electro-physiologic pattern of polarising and depolarising of the heart muscle. The popularity of ECG measurement stems from the fact that it is non-invasive, is very reliable, and conveys a large amount of information about the structure of the heart, function of electrical conduction system, condition of cells of the heart, chambers, among others, to clinicians and healthcare specialists. The effects of consumption of alcohol in bursts or over time in humans have been studied and documented extensively. Alcohol consumption affects the communication pathways in the brain, leading to changes in mood, behaviour, coordination, and ability to think clearly. Chronic alcohol consumption has also been shown to cause heart disorders such as cardiomyopathy, arrhythmias, stroke and high blood pressure, as well as liver ailments like alcoholic hepatitis , fibrosis, cirrhosis, and steatosis, as well as cause pancreatitis, weaken the immune system, and increase the risk of developing cancers of the mouth, esophagus, throat, liver and breast.

Heart Rate Variability (HRV) is the phenomenon of variation of the inter-beat interval or the time between successive R-R peaks of the PQRST waveform on a standard ECG signal. This variation in the inter-beat interval is a physiological phenomenon brought about by different inputs to the Sino-Atrial (SA) Node of the heart, which is where the cluster of cells which produce electrical impulses are located inside the heart. These inputs to the SA Node include the sympathetic nervous system (SNS) and parasympathetic nervous system (PSNS) as well as humeral factors. The SNS and PSNS are the two divisions under the autonomic nervous system, which performs unconsciously and is responsible for regulating bodily functions such as heart rate, et cetera. The SNS is known to prepare the body for intense physical activities, that is, activate the fight-or-flight response. This includes changes such as dilation of pupils and

increase in the heart rate. The preganglionic neurons originating from the spinal cord release acetylcholine, a neurotransmitter, at the synapse with the ganglia, which then activates the receptors on the postganglionic neurons, which in turn releases norepinephrine, which stimulates the adrenergic receptors on target tissues, and this is a response which primarily acts on the cardiovascular system, increases Heart Rate, and hence affects HRV. The PSNS relaxes the body and slows down or inhibits many high energy expending functions, and hence lowers the heart rate and blood pressure among other physiological changes. Hence, increased SNS activity and decreased PSNS activity results in decreased HRV. High Frequency (HF) activity (0.15 Hz to 0.4 Hz) is known to be associated with PSNS, whereas Low Frequency (LF) activity (0.04 Hz to 0.15 Hz) is accepted to be a combination of both SNS and PSNS activity.

Consumption of alcohol has been known to act as a depressant on the brain and nervous tissue. Several studies have linked chronic consumption of alcohol to changes in HRV, and these have detailed the correlation between the quantities of alcohol consumed as well as gender to extent of change in HRV. Hence, conclusions could be drawn about differences in alcoholic and normative test subjects based purely on HRV. HRV analysis consists of several methods, which are grouped under time-domain, frequency domain and non-linear. Time domain analysis focuses on the heart rhythm and its variations. Changes in time domain features gives an indication of magnitude of change in autonomic tone. Spectral analysis methods are now more sensitive or extracting more information regarding sympathetic and parasympathetic tone. For example, the power value of the HF content is considered as a measure of parasympathetic activity, while the power value of the LF content is reflective of both sympathetic and parasympathetic tone. The ratio of the LF component to that of the HF component is an index of sympathetic activity as well as balance between the sympathetic and parasympathetic nerves. The non-linear methods, which include the Poincare plot, reflect the autonomic function changes associated with long term consumption of alcohol. The plot not only delivers an outline, but also a detailed picture about the beat-to-beat behaviour of cardio-physiology. Studies have shown that chronic alcohol intake caused the decrease in Poincare plot indices accompanied with a decrease in area of the plot.

Auto Regressive modeling of the system also provides coefficients which can be used as features. These set of compiled features are used to train the

classifiers, which will be used to detect if a test subject is alcoholic or not. The classifiers chosen for comparison included Support Vector Machines (SVM) and Extreme Learning Machines (ELM). The Kernel trick was also applied on the dataset using Radial Basis Function (RBF) kernel. The uniqueness of this methodology lies in the usage of ARX coefficients as features, along with features provided from HRV analysis, and application of ELM to classification of test subjects as alcoholic or not based solely on ECG signals, it's performance compared to SVM, and construction of a simple, precise, hand held device to do everything from signal acquisition, pre-processing, feature extraction and classification, in real-time.

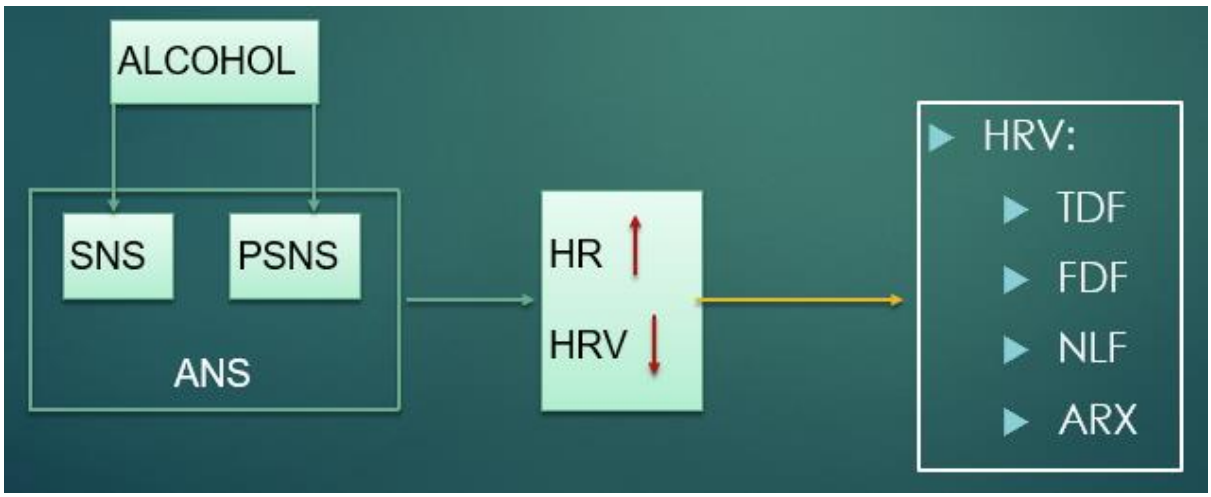


Table 1.1_1 Diagrammatic relation of alcohol consumption, its effect on physiology and HRV, to feature extraction using the HRV analysis

1.2 Problem Statement

Technology has advanced in various fields at rapid rates, however in an area that concerns the common well-being of humans, technology has remained dormant. Identifying accurately if a person is intoxicated is utmost important to keep public harm and nuisance at bay. The most common device used, the breathalyser has drawbacks that we aim to rectify. The common drawbacks of a breathalyser are:

i) Contamination of SiO_2 sensor requires frequent recalibration and replacement

ii) Breathalysers being a medium for propagation of contagious diseases

iii) Interfering components (like acetone) being higher in the breath of dieters and diabetics make them more prone to being falsely detected

iv) Infrared sensors detect the absorbance of the compound as a function of the wavelength of the beam when the infrared beam is passed through the sample breath chamber. The chamber is prone to environmental pollutants and aerosols leading to errors.

These disadvantages mentioned above are addressed with the use of infallible computers and well trained machine learning algorithms.
Heart rate variability [HRV] obtained from Electrocardiogram [ECG] is a useful biomarker and is used extensively in our paradigm to extract features. The features extracted are then used to train the system to classify patients as chronic alcoholic or otherwise. This may be useful in discriminating individuals based on their habits while preventing other external environmental conditions from altering or corrupts the readings.

1.3 Objective

The aim of this project is to:

1) Develop a prototype to read ECG signals
2) Extract features to perform HRV analysis
3) Classify the person under test as an alcoholic or otherwise, in real time.

1.4 Proposed Methodology

i. The first step is to study the advantages and disadvantages of HRV analysis over other markers for cardiovascular health and the effect of alcohol consumption on ECG signals.

ii. The next step is the familiarisation with the form and structure of the ECG signals, and understanding distortions (artefacts) that can occur in the signal. Two common artefacts seen in ECG signals are the wandering baseline and fuzzy (60 Hz) distortions caused due to probe movement, uneven conductive gel, muscle movement, etc.

iii. A survey of available devices for measuring and recording ECG signals is done. The device made by a previous student connects to a phone via bluetooth and the data is then routed to a server from the phone. This device was found to be noisy and required extra software like Audacity for filtering

purposes and hence was not real time. It was proposed to interface the sensor directly with the Raspberry Pi 2 and do all the processing on it, making the overall system more compact and real time.

iv. Parallel to working on the sensor hardware, once the ECG waveforms are understood, work with the training data set is started to decide the feature extraction techniques and classifiers that could be implemented. Survey of tools like KubiosHRV to understand time domain and frequency domain feature extraction for HRV analysis is done. The training data set is obtained from Autonomic Lab, Department of Neurophysiology, NIMHANS and consists of 38 samples of alcoholic and 29 samples of normative subjects.

v. To this data, preprocessing is performed to remove artefacts, and then time domain, frequency domain, and non-linear methods are used for feature extraction from the ECG signals. On extracting the desired features, ELM and SVM are classifiers that are chosen to be trained using the dataset. Kernel functions are also tried in order to check if their usage might improve accuracy.

vi. Accuracy of the classifiers is checked using leave-one-out cross validation and k-fold cross validation, and classifiers providing highest consistent average accuracy in these validation methods is selected.

vii. Based on the algorithms finalised on, the controller board on which the feature extraction and classification algorithms will be implemented are chosen. The target device is the Raspberry Pi 2 currently, as it has a 900 MHz quad-core CPU that is necessary for machine learning applications.

viii. Training of the classifiers to obtain weights is done on MATLAB and the trained parameters are used on the Raspberry Pi 2.

ix. Finally, the sensor is interfaced with the Raspberry Pi 2, which is uploaded with trained parameters, ready to classify real time ECG data.

CHAPTER – 2

LITERATURE SURVEY

This chapter covers literature survey performed in order to understand the effect of alcohol consumption on HRV, the methods involved in HRV analysis, the features that can be extracted using these methods and the classifiers which can be used for the dataset obtained. It also covers the literature studied for similar implementations, to find out which features were extracted, which classifiers were used and what the accuracies obtained were. Several papers were studied for design of the sensors used in order to record ECG signals and documentation of the micro controllers that could be used to implement the real time device.

2.1 Electrocardiogram, Noise, Artefacts and Signal Acquisition

The Electrocardiogram provides the healthcare specialist a lot of information about the heart, understand and come to conclusions about any underlying factors which result in variations observed in the waveforms. The ECG signal is captured by placing several electrodes on the skin of the test subject at the limbs and on the chest and measuring the magnitude of the potential at these points. There can be a standard 10 electrode ECG as well as a three electrode ECG in which the three electrodes are placed at the points shown by Einthoven's triangle. The three electrode ECG measures the potential difference between the different electrodes. This potential difference is of the order of a few hundred microvolts to a few millivolts, and hence needs to be amplified greatly using amplifiers. The signal is also affected by noise, from sources like power sources, environment and static electric charges, and artefacts, caused due to muscle movement and breathing. These need to be filtered out using hardware and software filters like high pass filters, low pass filters, notch filters and IIR filters. Several designs for ECG signal acquisition were studied and adopted in designing our circuit, and these can be found in [1], [2], [3], and [4].

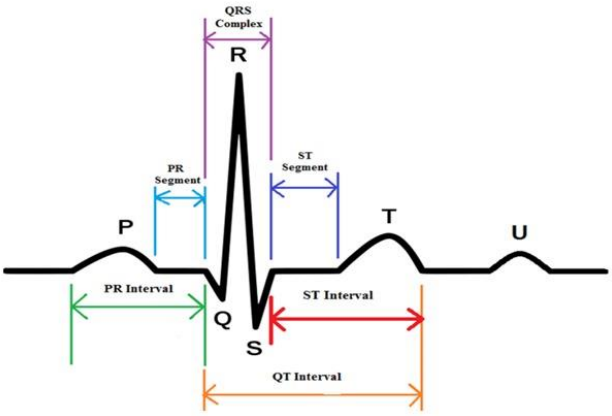


Fig 2.1 Schematic Representation of ECG Waveform

2.2 Effect of Alcohol on Heart Rate Variability

The ECG waveform consists of PQRST complexes, which represent the polarization and depolarization of the chambers of the heart. The R peak in the waveform represents the depolarization of the right and left ventricles of the heart. This is the portion of the PQRST complex that is most striking, and hence can be detected easily. The time interval between successive RR peaks is known as inter-beat-interval. HRV analysis involves finding variations in these inter-beat-intervals. Studies have shown that alcohol consumption affects autonomic nervous system activity, which includes the sympathetic and parasympathetic nervous system, which in turn affect the activity of the heart. HRV analysis has been performed on alcoholics for different amounts of alcohol consumption and for time intervals after the subsequent consumption, for males and females, and it was concluded in [5] that HRV decreases for test subjects after ingestion of alcohol, and the effect of decreased HRV was less pronounced in females than it was in males. HRV and its various applications in assessing activities of autonomic nervous system are comprehensively detailed in [6].

In [5], electrocardiograms of 1,000 chronic alcoholic patients were taken and analyzed to find evidence that excessive consumption of alcohol may produce changes in the electrocardiogram. The predominant abnormalities that were observed by the authors were sinus tachycardia and nonspecific T-wave changes. Studies conducted as shown in [7] is one of the first few studies that emphasized the need to study the relation between not the just alcohol consumption and the liver but also consider the effect of alcohol consumption

on the heart. Various changes were seen in the ECG of alcoholic patients like the dimple T wave, spinous T wave, cloven T waves, etc. This study clearly establishes the fact that alcohol consumption leads to variation in heart beat.

A direct correlation between moderate to heavy alcohol consumption over time and heart rate variability, difference in HRV between the genders, effect on test subjects during exercise, effect of alcohol consumption on the autonomic nervous system and hence the effect on HRV and ECG signals obtained has been observed and documented in [5], [8], [9], [10], and [11].

2.3 HRV Analysis and Features Extracted

Now that sufficient evidence had been obtained about the effect of alcohol consumption on HRV, the focus shifts towards extracting features from the ECG signal. Several softwares that performed HRV analysis were looked into, which included KubiosHRV, HRVAS (Heart Rate Variability Analysis Software), NerveExpress and IntelleWave, out of which KubiosHRV was chosen for further study since it was available as freeware, was lucid and elaborative. The documentation for KubiosHRV [12] listed out all the methods used for HRV analysis, which are broadly grouped into time domain methods, frequency domain methods and non-linear methods. The documentation also detailed the features that were extracted and methods of extracting these features from the signal.

The time domain feature extraction methods first involve obtaining the R-R intervals. The frequency domain feature extraction methods involve calculating the power spectral density. Non-linear feature extraction methods utilises Poincare plots, et cetera. Along with these features, in order to obtain better classification accuracies, exogenous auto-regressive modelling was also looked into in order to obtain more features, and the system identification toolbox was identified to be an aid in this process, as is described in [13], [24], [14].

2.4 Classifiers and Similar Implementations

In [15] the authors collected ECGs from 50 volunteers. They applied pre-processing for noise suppression and signal segregation into a number of samples, where each sample represented the heart activity for one heartbeat. To identify characteristics of an alcoholic, features such as Pmax, Pd, means and variances of P, R, S waves and R-R intervals are extracted. The paper used SVM as a classifier and 10-fold cross validation. This provided sufficient encouragement for us to go ahead with using our dataset and try different classifiers for the same, try to obtain higher accuracies, and implement the real time device.

[16] gives a comprehensive view of various supervised machine learning classification techniques and provides interesting domains where machine learning can be applied.

In [17], the author describes the architecture of ELM, how it is different from other neural networks which use back propagation to iteratively tune parameters, and compare it with other classifiers like SVM and Least Squares SVM (LS-SVM).

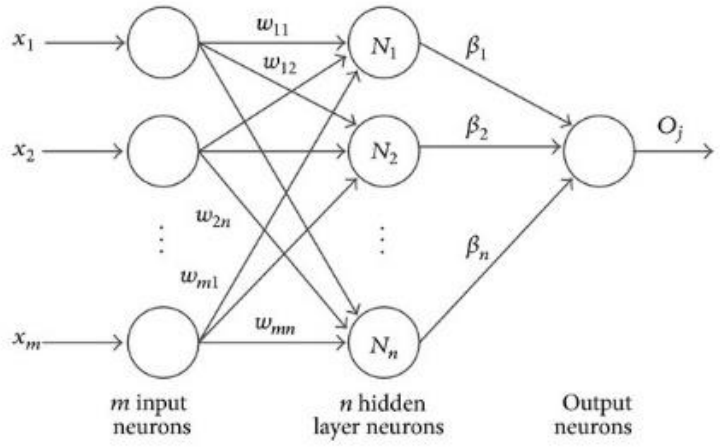


Fig 2.2 Architecture of an Extreme Learning Machine

In [18], the authors describe an algorithm to build the structure of the ELM neural network during the training phase, instead of fixing the architecture a-priori. This would optimise the number of hidden layer neurons in an intelligent way, instead of using brute force to find the number of hidden layer neurons that gives best accuracy without over fitting.

[19], [20] provide us with an understanding of the Support Vector Machine learning algorithm, and the also talks about Kernels, which allow us to vary the dimensionality of the feature dataset, and the SMO algorithm, which gives us an efficient implementation of SVMs.

CHAPTER – 3

METHODOLOGY

3.1 INTRODUCTION

This section describes the overall working of the ECG classification project. The chapter is divided into two sections (Section 3.2, Section 3.3, and Section 3.4), and gives details about the hardware, software, and real time portions of the project.

3.2 HARDWARE

This section of the chapter covers topics about the hardware aspects of the project. The idea was to make a portable handheld device that can record a person's ECG signal and detect if the person is a chronic alcoholic. To have sufficient computation power and still be portable, a Raspberry Pi has been used. To the Raspberry Pi, a sensor to amplify and filter the ECG signal is attached. This chapter covers details about the Raspberry Pi, the self-designed ECG sensor and the readymade ECG sensor (AD8232).

3.2.1 ECG Sensor Circuit Design

To begin the design of the ECG sensor, first, details about the human physiology were studied. This was a necessary step to figure out where to place the ECG probes and to estimate what amplitude of signal could be expected from the probes. There are several places on the body where the probes can be placed as shown in Fig 3.1. The requirement of this project was portability and ease of use. Thus the differential voltage across the left and right index finders was calculated. Once the probe placement was studied and differential signal voltage was estimated, the possible sources of noise in ECG signals were studied. This was done to ensure that the correct filtering specifications were known before starting the actual design of the sensor and filters. Finally, the block diagram level of the sensor design was understood and the complete ECG sensor and filters were designed.

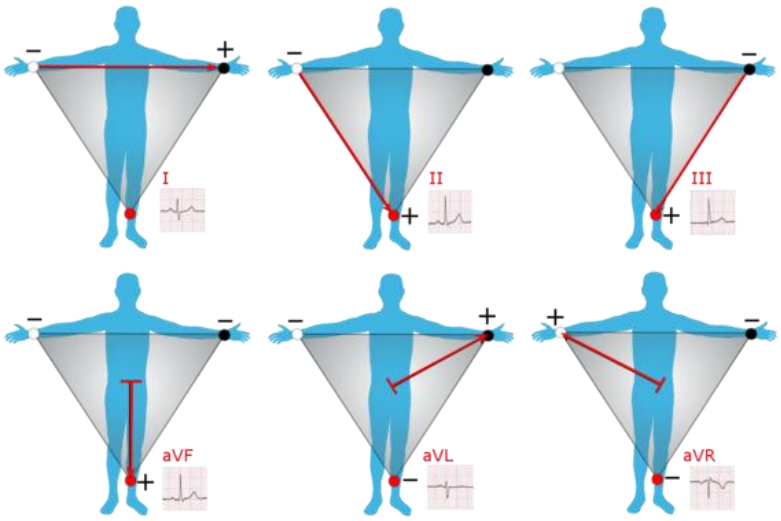


Fig 3.1 Traditional placement of ECG probes

From [1] it was noted that ECG signals can be detected only in the milli-volts at the surface of the skin. This meant that an amplifier would be required to bring the signal level to the order of a few volts, so that signal filtering and meaningful signal processing could be performed on the signals for feature extraction. The ECG signal being a low amplitude signal, required an very accurate, low noise amplifier. A suitable such amplifier which was designed for biomedical applications was the AD620. An ECG signal sensor circuit was provided in the datasheet of the AD620. This same circuit was tested on multisim and on obtaining good amplification this amplifier was procured and the hardware circuit was realized on a breadboard. The circuit built is given below (Fig. 3.2):

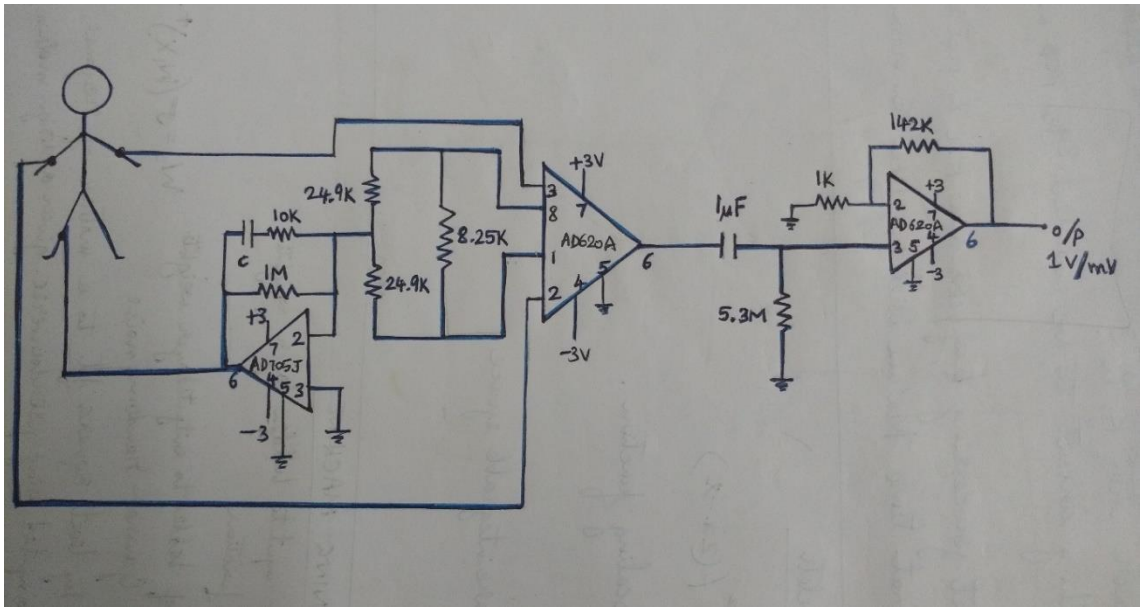


Fig 3.2 Circuit Design-1

The two stage amplification circuit that was made on the breadboard did not seem to work as a complete unit. The virtual ground and first stage amplification sections of the circuit worked fine, however signals got at the output of the second amplifier were erroneous in shape and amplitude. The HPF present at the junction of the two amplifier was dropped in hopes that proper two stage amplification could be achieved, but that did not solve the issue either. In the process of experimenting with the two stage amplifier, it was seen that the AD620 provided high levels of accurate amplification of signals and a single stage amplifier would be sufficient. Thus, by changing the Rg resistors being supplied to the IC, a single stage amplifier was designed. Rg of 470 ohms was used to obtain a gain of 1000. A gain of 1000 was aimed for, to enable voltages in the mV range to be amplified to the volts range.

$$Gain_{AD620} = 1 + \frac{49.9k\Omega}{R_g} \qquad \text{(eqn 3.0.1)}$$

Some reading up on the purpose and need for virtual ground showed that virtual grounds are used when only a single polarity of voltage supply is available but two polarities of voltage supply is required. The circuit that was planned to be developed needed to be portable and a 9V battery was sufficient to power the amplifiers. It was decided to simplify the circuit further by removing the virtual ground circuit and use two 9V batteries to create a two-polarity source.

Now the two-stage amplifier circuit had been reduced to a single stage amplifier and the virtual ground circuit had also been removed. This posed a problem to the filtering that was being done to the signal as the HPF that was present at the junction of the two amplifiers was removed. This required the use of filters at the output of the single stage amplifier. Sources [4] and [3] showed that a HPF and an LPF of 100Hz and 0.05Hz values respectively would be needed at the output of the amplifier.

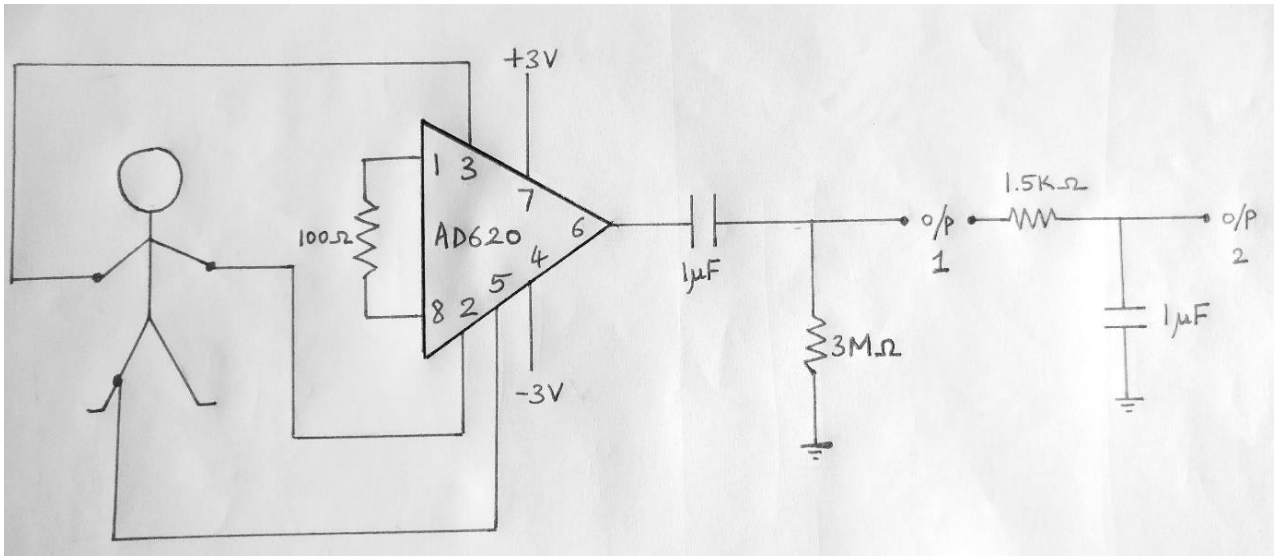


Fig 3.3 Circuit Design-2

The circuit above (fig 3.3) gave output a signal as shown figure (Fig 3.4) when the probes were connected to the limbs. The signal did not seem to have meaning. However, when the probes were detached, there was a clean signal as seen in Fig 3.5. It was noticed that this signal had exactly 50Hz of frequency, which is the same frequency as AC sources in Asia. This realization showed that Fig 3.5 was actually amplified 50Hz power-line noise with the ECG signal being superimposed on it.

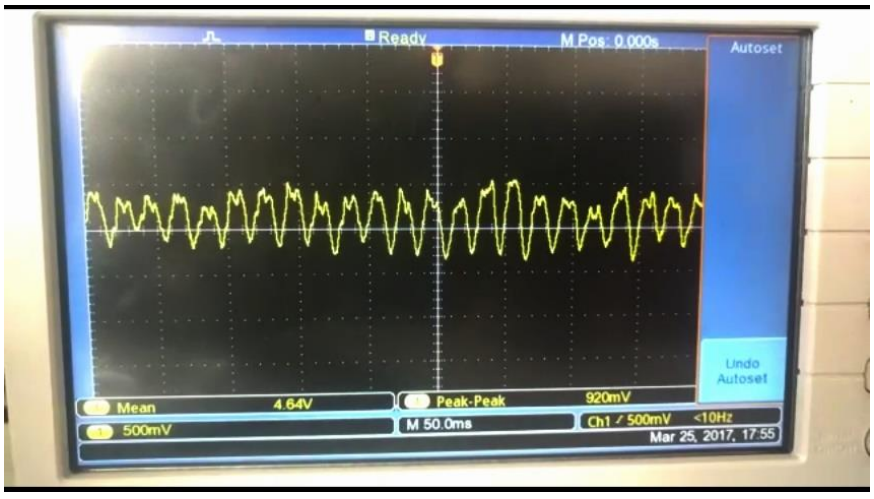


Fig 3.4 Superposed Output – with Probe Connection to Limbs

Fig 3.5 50Hz Output without Probes Connected to Limbs

The goal for the next circuit was to remove the 50Hz signal components and retain on the ECG signal. To do that, a twin T notch filter was designed with variable Q factor. References [21] and [22] were used to design the notch filter. The circuit designed is as given in (Fig 3.5) and the calculations made to obtain a 50Hz band rejection were as given by (eqn 3.0.2). The Q factor of the notch filter was set to around 0.75 by varying the values of R6 and R7 according to (eqn 3.0.3).

$$f_{notch} = \frac{1}{2\Pi RC} \qquad \text{(eqn 3.0.2)}$$

$$1 - \frac{1}{4Q} = \frac{R_7}{R_7 + R_6} \qquad \text{(eqn 3.0.3)}$$

The R_g used was 100Ω in order to obtain an amplification of about 1000. The R and C were set to $3.3k\Omega$ and $1\mu F$ to obtain a cutoff frequency of 50Hz. Q was set to 0.75 by using R_6 and R_7 as $1k\Omega$ and $2k\Omega$ respectively. All modifications resulted in the circuit (Fig 3.6) below:

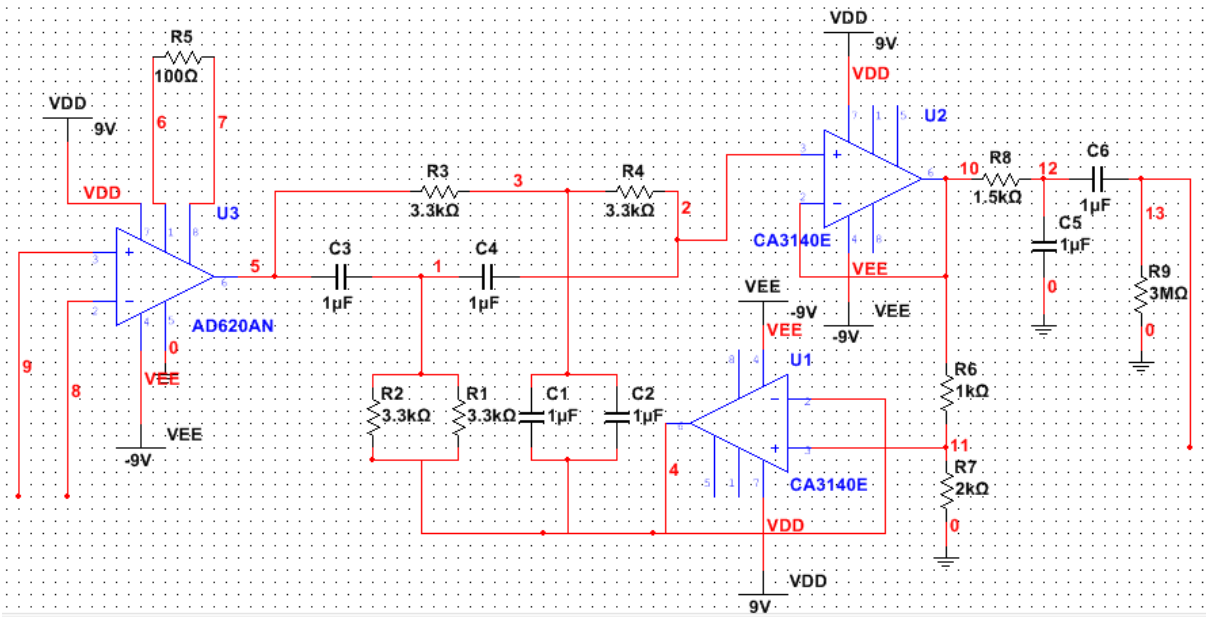


Fig 3.6 Circuit Design-3

The above circuit was tested on a breadboard and a pulsating signal of approximately 1Hz was obtained on the CRO. However as the circuit was made on the breadboard, slight disturbances to the wires caused a loss in the output signal. To counter that, the same circuit was soldered onto a breakout breadboard/PCB. Figure (Fig 3.7) shows the soldered notch filter circuit.

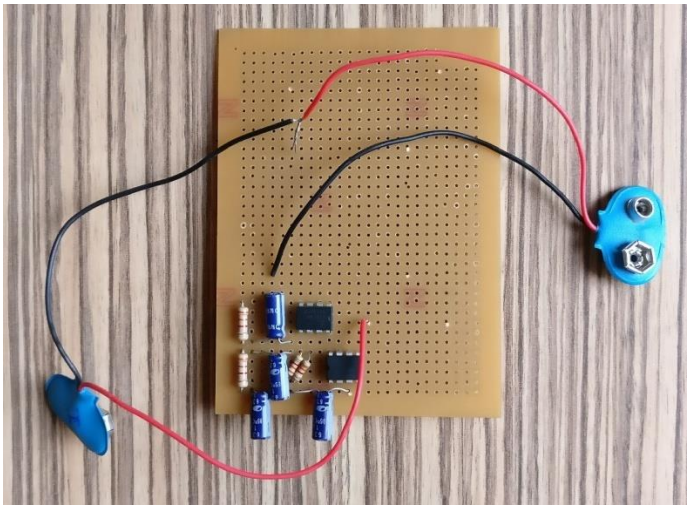


Fig 3.7 Soldered Circuit

The entire circuit containing the AD620 and the HPF/LPF could not be soldered within the course of this project. However, in order the serve the greater overall need of the project which was to acquired ECG signals of subjects and analyse the signals for classification purpose, a market ready ECG sensor, the AD8232

was procured and used. Details about the AD8232 and its integration with the Raspberry Pi are given in the section (Section 3.2.5) that follow in this chapter.

3.2.2 Heart Rate Monitor – AD 8232

While a large amount of research had been done on developing a sensor from scratch, it was not a part of the initial scope of the project. While the notch filter showed promising signals on the breadboard for a short duration of time, the soldered version could not be tested fully. To ensure the completion and validation of the alcoholic classification project on the whole, a readymade ECG sensor was procured. Details about the sensor hare mentioned in this section.

The Heart Rate Monitor is used to measure the electrical activity of the heart. It uses AD8232 as its core chip which is one of the popular integrated signal conditioning block for ECG and other bio-potential measurement applications.

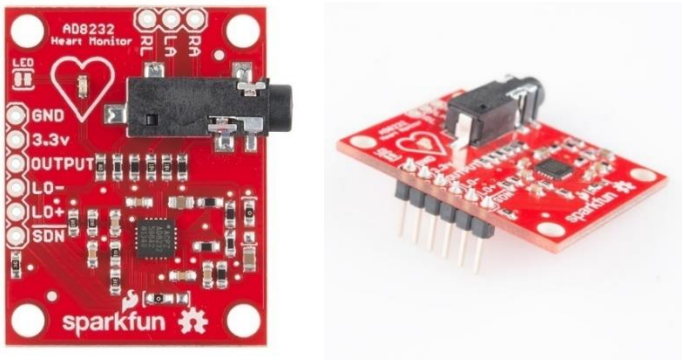


Fig 3.8 AD8232

The AD8232 Heart Rate Monitor breaks out six essential connections from the IC SDN, LO+, LO-, OUTPUT, 3.3V, GND for operation. The Board also has RA(Right Arm),LA(Left Arm) and RL(Right Leg) pins for custom application. Additional features are listed below:

1. Analog Output
2. Leads-Off Detection
3. Shutdown pin
4. LED indicator
5. 3.5mm Jack for Biomedical Pad connection

AD8232:

At the core of the Heart Rate Monitor is the AD8232. It is one of the most popular and widely used IC for extraction and filtering of bio-potential signals in various bio-medical applications is the AD8232. The features that make AD8232 appropriate for these applications are:

1. The AD8232 implements a two-high pass filter for eliminating motion artefacts which often affect the small bio potential signals.

2. Three pole low pass filter: The implementation of this filter allows the IC to remove additional noise.

3. Fast restore function: In many applications due to the low cut-off frequency used in the high pass filters, signals may require longer settling time. This long settling time could cause an unwanted delay in obtaining the signal. Hence this function reduces the duration of long settling tails of the high pass filters.

4. Specialized instrumentation Amplifier: The IC has contains a multi-purpose Instrumentation Amplifier which amplifies the ECG signal while rejecting the electrode half-cell potential on the same stage.

5. Right leg drive amplifier: The common mode output at the instrumentation amplifier is inverted by the right leg drive (RLD) amplifier. When the right leg drive output current is injected into the subject, it counteracts common-mode voltage variations, thereby improving the common-mode rejection of the system.

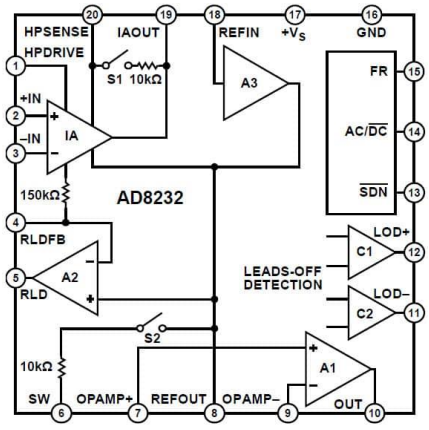


Fig 3.9 Internal Pin Diagram of AD8232

All the above feature allows AD8232 to extract, amplify, and filter small bio-potential signals in the presence of noisy conditions.

3.2.3 Rapsberry Pi

The Raspberry Pi is a series of small single board computers developed by the Raspberry Pi Foundation. These boards are approximately credit-card sized and have a standard mainline form-factor.

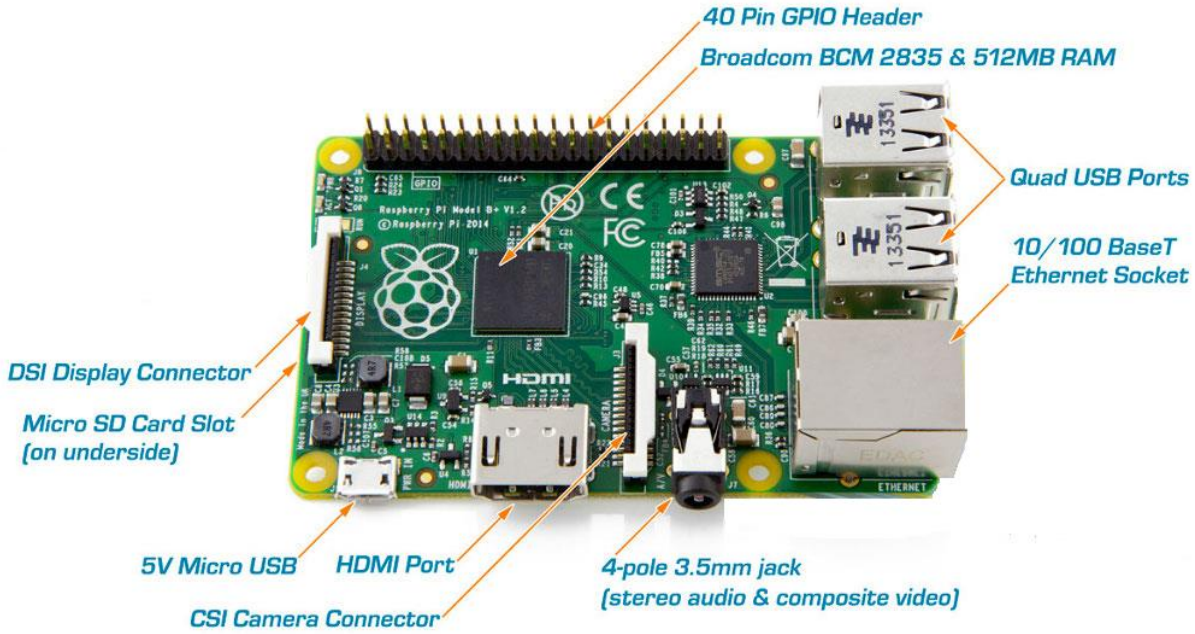


Fig 3.10 Raspberry Pi 2 model B

Several generations of Raspberry Pi have been released. The version used in this project is Raspberry Pi 2 Model B.

All models of Raspberry Pi have a Broadcom system on chip (SoC) which includes an ARM compatible central processing unit (CPU) and on chip graphic processing unit (GPU).

The specifications (specs) of Raspberry Pi 2 are:

1) Processor: Broadcom BCM2836 SoC with a 900 MHz 32-bit quad-core ARM Cortex-A7 processor with 256 KB shared L2 cache.
2) Random Access Memory(RAM): 1 GB of RAM
3) Peripherals: 17 General purpose Input Output(GPIO) plus specific function pins have been provided

4) Network: There is separate Ethernet port provided to support 10/100Mbps
5) There are four additional Universal Serial Bus(USB) provided for connecting peripherals like keyboard, mouse
6) Storage: There is a separate Secure Digital(SD) card slot provided
7) Power source : 5V via Micro USB or GPIO header

3.2.4 Analog to Digital Converter (ADC)

Analog signals when stored on a computer are stored in their digitized format. The Raspberri-Pi does not have a built in analog to digital converter (ADC). This means an external chip performing the ADC will be necessary. The MCP3008 is the ADC that has been interfaced with the Raspberri-Pi.

Analog to Digital convertor is a system that converts an analog signal into a digital signal. There are various ADC available in the market of which MCP3008 is used in this project. It is a 10-bit ADC with on board sample and hold circuitry.

Some of the features of MCP3008 are:

1. 10 bit resolution
2. 8 input channel
3. Serial Peripheral Interface (SPI) serial interface
4. High sampling rate
   200 kilo samples per second (ksps) at a supply voltage of 5V and 75 kilo samples per second (ksps) at a supply voltage of 3.3V

Fig 3.11 MCP 3008 ADC

3.2.5 Integration of hardware:

The previous section gave specifications about the different components used in the hardware portion of the project. This section describes how all of the hardware was integrated.

3.2.5.1 Setting up Raspberry Pi

(a) SD card setup

Raspberry Pi 2.0 does not come with a pre-built operating system. Instead the operating system has to be flashed on the SD card which is then inserted into the Raspberry Pi. The operating system on SD card is installed by using the following steps:

i. SD card is inserted into the SD card reader.
ii. Win32DiskImager utility is downloaded from the Sourceforge Project page as an installer file, and the software is installed by running it.
iii. The Operating System (OS) of choice is downloaded from official raspberry site. For this project Raspbian Wheezy was used.
iv. The image of the OS is then extracted from the downloaded file.
v. The Win32DiskImager utility software installed is then opened.
vi. The extracted image is then selected and 'Write' button is pressed to write the OS to the SD card.
vii. On successful completion of the process the SD card is ejected and inserted into the Raspberry Pi.
    a. The Raspberry Pi is ready for use with Raspbian Wheezy installed.
    b. ##The Raspberry Pi is then connected to the Internet using Ethernet cable.

(b) Internet Protocol (IP) address of Raspberry Pi (R-Pi)

In order to work with the Raspberry Pi the IP address of Pi is needed. There are two ways to obtain this:

i. Working with Rpi using Television (TV) mode:
In this method the Rpi is connected to a TV screen using HDMI cable. The Rpi is connected to a router using Ethernet connection for Internet connection.

ii. The IP address is then obtained by typing in “ipconfig” in the terminal window of the Rpi.

iii. Working with Rpi in “headless” mode :
In this case a static IP address is assigned by the user. This is the approach used in this project as it is not always convenient to have a TV screen. The steps to set the static IP address is as follows:
(a) SD card is removed from Rpi and then inserted into the SD card reader.

(b) The SD card contents are then opened.

(c) A file named “cmdline.txt” is opened using Notepad++

(d) At the end of the file the following line is appended:
Syntax: ip=<client-ip>:<server-ip>:<gw-ip>:<netmask>:<hostname>:<device>:<autoconf>
Example:
ip=169.254.3.14::169.254.56.85:255.255.0.0:rpi:eth0:off

(e) The file is then saved and SD card is safely ejected from the system and inserted back into the Raspberry Pi.
The Rpi is then connected to a laptop using Ethernet

(c) Communicating with the Rpi

SSH protocol is used for establishing the communication between the Rpi and the laptop. For this a free and open-source terminal emulator called PuTTY is used. PuTTY supports many variations on the secure remote terminal, and provides user control over the SSH encryption key and protocol version.

3.2.5.2 Connecting Heart Rate Monitor to RPi using ADC MCP3008

The following is the circuit diagram required for the setup:

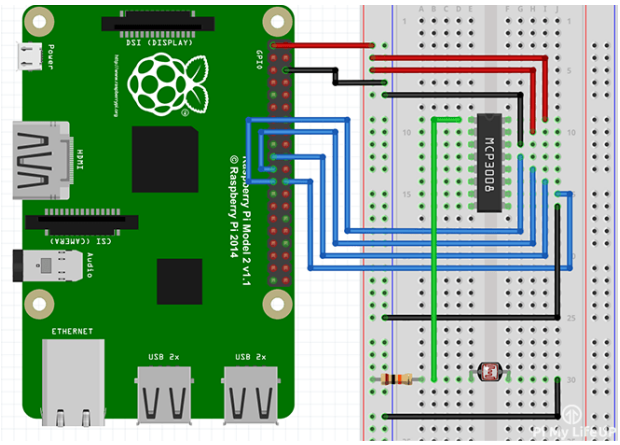


Fig 3.12 MCP 3008 interface to Raspberry Pi

First a 3v3 pin is connected to the positive rail on the breadboard and a ground pin to the ground rail on the breadboard. The following connections are made:

MCP3008	Rpi
VDD (Pin 16)	3.3V
VREF (Pin 15)	3.3V
AGND (Pin 14)	GROUND
CLK (Pin 13)	GPIO11 (Pin 23/SCLK)
DOUT (Pin 12)	GPIO9 (Pin 21/MISO)
DIN (Pin 11)	GPIO10 (Pin 19/MOSI)
CS (Pin 10)	GPIO8 (Pin 24/CE0)
DGND (Pin 9)	GROUND

Table 3.1 Connections between MCP3008 and Raspberry Pi

3.2.6 Communication protocols

With so many components being used for the hardware, on integrating all of it, some form of communication between all the components was required. Following are the two communication protocols used in this project:

1. Serial-Peripheral Interface (SPI) protocol: This communication protocol was used between the Raspberry-Pi and the MCP3008.

   SPI is a synchronous data bus used to send data between microcontrollers

and small peripherals such as shift registers and sensors. Here synchronous data bus means that it uses separate clock and data lines. The clock is an oscillating signal that tells the receiver exactly when to sample the bits on the data line. This can be the rising or falling edge of the clock signal.When the receiver detects that edge, it will immediately look at the data line to read the next bit.

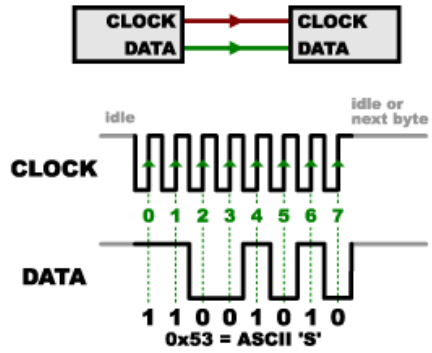


Fig 3.13 Synchronous Data Bus

In SPI, only one side generates the clock signal.The side that generates the clock is called the “master”, and the other side is called the “slave”. There is always only one master which is the microcontroller,but there can be multiple slaves.

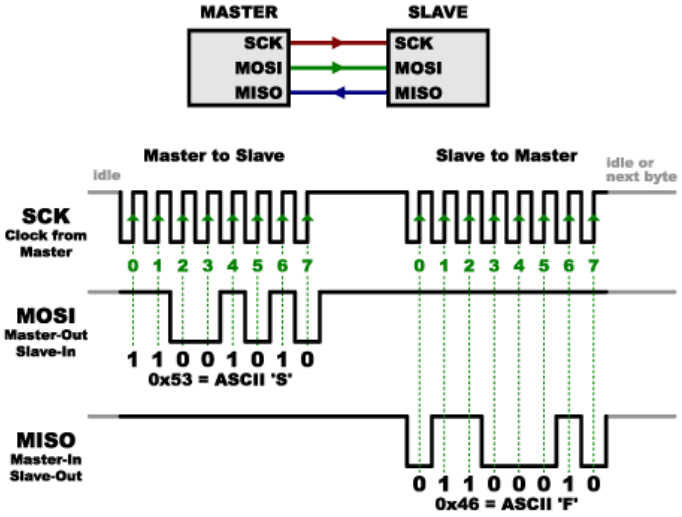


Fig 3.14 Sending and receiving data using SPI

When data is sent from the master to a slave, it is sent on a data line “Master Out / Slave In”(MOSI). If the slave needs to send a response back to the master, the master will continue to generate a prearranged

number of clock cycles, and the slave will then put the data onto a third data line called "Master In / Slave Out"(MISO).

Advantages of SPI:

(a) It is faster than asynchronous serial
(b) The receive hardware can be a simple shift register
(c) It supports multiple slaves

2. Secure Shell: This was the form of connection that was used between the Raspberry-Pi and the computer while setting up the raspberry pi and running various python scripts.

   Secure Shell (SSH) is a widely used cryptographic network protocol for operating network services securely over an unsecured network. The best application of it is for remote login to computer systems by users. SSH provides a secure channel over an unsecured network in a client-server architecture, connecting an SSH client application with an SSH server.

   Some popular uses of SSH are:

   (a) For login to a shell on a remote host
   (b) For executing a single command on a remote host
   (c) For setting up automatic login to a remote server
   (d) Secure file transfer

3.3 SOFTWARE

In this section details about the dataset used, pre-processing performed on the dataset, features extracted from the dataset, feature reduction performed and classification algorithms trained has been provided. While pre-processing and feature reduction were important steps in obtaining a well trained classifier, the bulk of the software end of the project focusses on explaining feature extraction and classification algorithms. Details about the various types of features extracted has been explained in section 3.3.3, and some background information about the two classifiers used on the dataset has been provided in sections 3.3.5 and 3.3.6.

3.3.1 Dataset Description

In order to classify test subjects as alcoholics or normative, with a reasonable accuracy, the classifier has to be trained with several samples of the test subjects, and subsequently tested with more samples. These samples along with the appropriate labelling constitute the dataset.

The data in the dataset is an array of values which represent the ECG signal of the test subject. The ECG data used has been recorded at the Autonomic Lab, Department of Neurophysiology, NIMHANS, Bengaluru. The data was recorded after taking informed consent adhering to Helsinki's declaration. At the Autonomic Lab , HRV is done using proprietary hardware and software setup by AD instruments, Australia. The product used in the lab premises for observing and recording HRV is PowerLab. This device uses high sample frequency (of the order of 1kHz) to record electronic activity of the heart and other signals pertaining to other functions such as respiratory functions et cetera. The raw ECG data was extracted as five minute samples in European Data Format [EDF]. The dataset comprises of 67 samples, out of which 38 are ECG recordings of alcoholic test subjects and 29 are of normative test subjects.

3.3.2 Pre-processing

ECG data acquired from a patient contains various types of disturbances and noise, which makes it difficult to extract features from the signal. The most common types of disturbances are [23]:

a. Power-line Interference: ECG signals are measured as the differential voltage that exists between two points on the body and turn out to have very small voltage amplitudes (in the mV range). The small voltage amplitude of ECG signals makes it susceptible to interference from AC signals present at any point in the circuit. The primary source of power-line interference is the AC power-line used as the power source for the ECG recording devices and display monitors or CROs. The frequency of the signal in the power-line in India is 50Hz, leading disturbances on the ECG signals also with the same frequency.

b. Baseline Wandering: Gradual changes in the skin impedance of the patient and the patient's breathing lead to low frequency baseline wandering. Baseline wandering is seen as an overall rise and fall of the PQRST complexes of the ECG signal.

c. Motion Artefacts: Movements by the patient or the electrode cause mild to severe disturbances in the baseline of the ECG signal. While mild movements like breathing or slight movement of limbs not connected with the probes do not completely 'submerge' the ECG signal, abrupt movements by the patient can completely mask the QRS complex with meaningless noise.

In the dataset used, all three forms of artefacts are observable and have been dealt with to obtain cleaner waveforms for feature extraction.

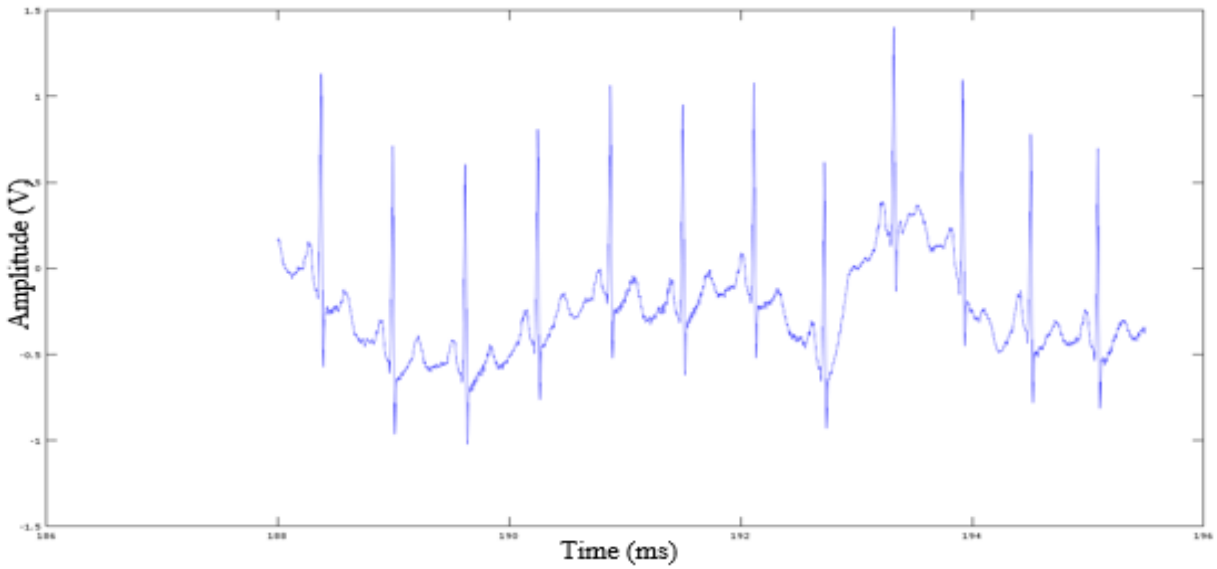


Fig 3.15 Baseline Wandering and Powerline noise in ECG

3.3.2.1 Infinite Impule Respose (IIR) Filtering

Different filtering techniques have been used to obtain suitable signals for time domain, frequency domain and non-linear frequency extractions. Time domain and non-linear feature extraction rely primarily on the RR intervals of the ECG signal. Thus, we focus on filtering out artefacts and other sections of the ECG sequence while maintaining a high amplitude for the QRS complex. Such results have been obtained using a first order, low pass butterworth filter, having

a pass band attenuation of 0.2 and cut-off frequencies of 5Hz and 7Hz. The filter was designed for values mentioned previously since, [12] indicates motion artefacts to be present in the 0Hz to 5Hz range. The same filter also helped rid the signal of baseline wandering. Removing the motion artefacts and the baseline wandering essentially ensured that the required R peaks remained the prominent portion of the signal. Finally, the RR intervals were obtained by setting a threshold and measuring the time difference between the occurrences of maximas in those sections of the signal that crossed the threshold. A simple first order IIR filter sufficiently filtered the signal to make the R peaks prominent and allow the calculation of RR intervals.

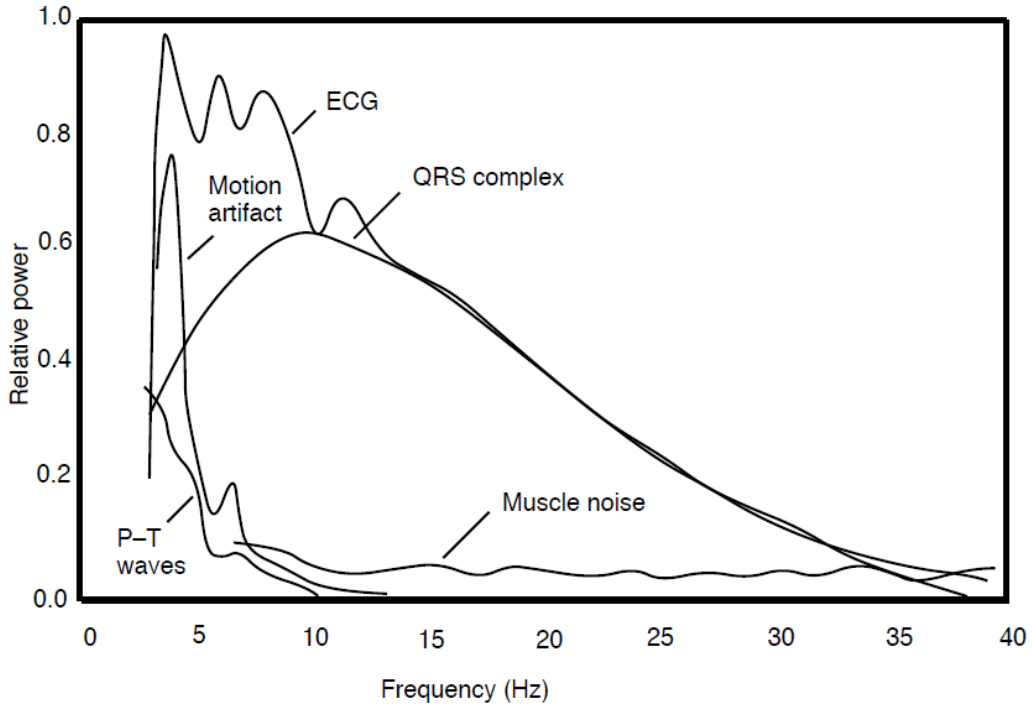


Fig 3.16 PSD of a sample ECG signal with noise components

The red line in the graph (fig 3.17) below represents the threshold that was used to detect the peaks. The threshold value was set at 70% of the total voltage swing occurring in the ECG signal.

The same IIR filtered signal was not sufficient clean to accurately extract frequency domain features, as it still contained some low amplitude noise at its baseline. Frequency domain features are obtained primarily by taking the power spectral density (PSD) of the signal over different frequency ranges. To obtain

consistent PSD values and accurate frequency domain features, a clean signal is required. Wavelet Transform has been used to achieve high precision filtering.

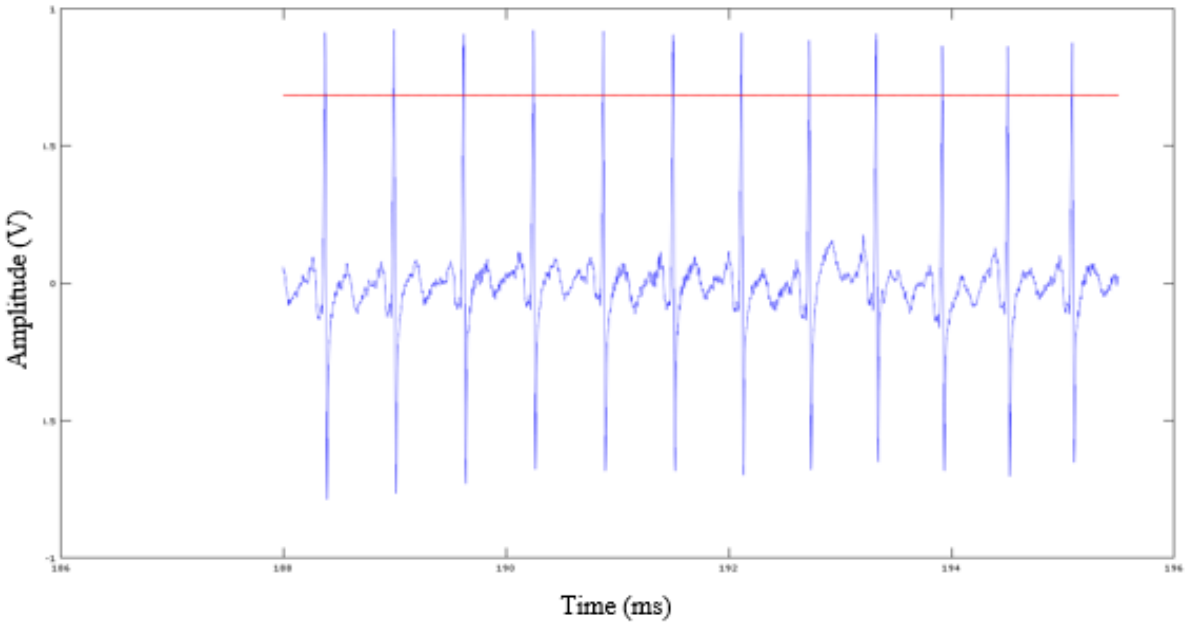


Fig 3.17 IIR Filtered Signal without Baseline Wandering

3.3.2.2 Wavelet Decomposition

One of the most well know signal analysis tools to obtain frequency information of the signal, is the Fourier Analysis. Fourier Analysis breaks the signal down into sinusoids of different frequencies.

Fig 3.18 Diagrammatic Representation of Fourier Transform applied to a Signal

However, Fourier Analysis has a serious drawback that when a signal is transformed from time domain to frequency domain, the time information is lost. This drawback is not important for stationary signals. But most real world signals contain numerous non-stationary or transitory characteristics: drift, trends, abrupt changes, and beginnings and ends of events.
Another analysis method that overcomes the drawback of Fourier Transform is the Short Fourier Transform (STFT). The STFT is a compromise between the time and frequency-based views of a signal. A signal is mapped into two dimensional function of time and frequency.

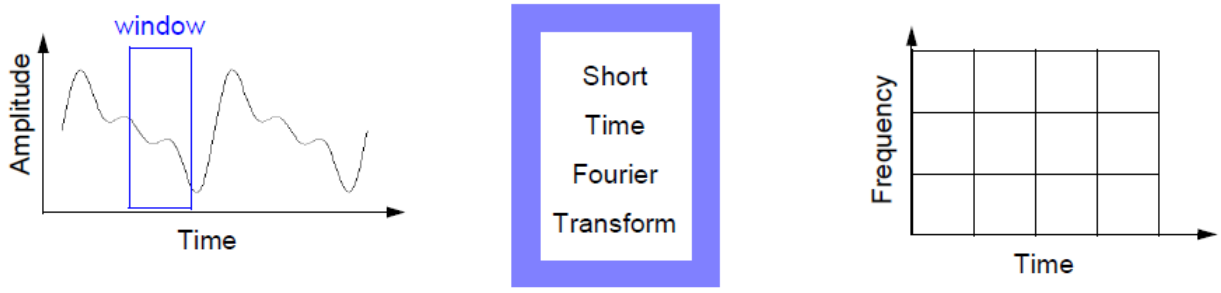


Fig 3.19 Diagrammatic Representation of

Short Fourier Transform applied to a Signal

The drawbacks of STFT are:

(a) Limited precision depending on the size of window

(b) Once a particular size for the time window is selected, that window is the same for all frequencies

Hence there is a need for an analysis technique that represents a windowing technique with variable-sized regions. Wavelet analysis allows the use of long time intervals where more precise low frequency information is needed, and shorter regions where high frequency information is needed.

Wavelet analysis is the breaking up of a signal into shifted and scaled versions of the original wavelet. A wavelet is a waveform of limited duration that has an average value of zero.

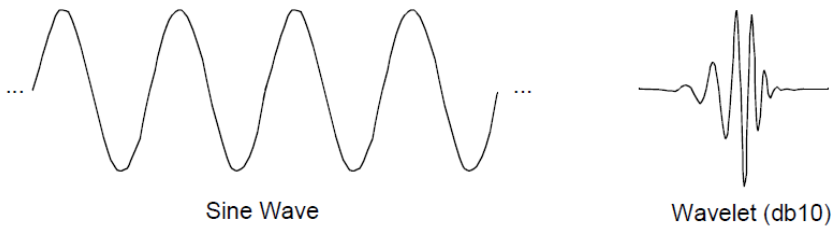


Fig 3.20 Sine wave and a Mother Wavelet

Depending on the application, different types of wavelet transform tools are used. They are:

1. Continuous Wavelet Transform (CWT)
   The continuous wavelet transform (CWT) is defined as the sum over all time of the signal multiplied by scaled, shifted versions of the wavelet function ψ:

$$C(scale, position) = \int_{-\infty}^{\infty} f(t)\psi(scale, position, t)dt$$

   The output of the CWT are many wavelet coefficients C, which are a function of scale and position. These coefficients on multiplication by scaled and shifted wavelets yield the constituent wavelets of the original signal.

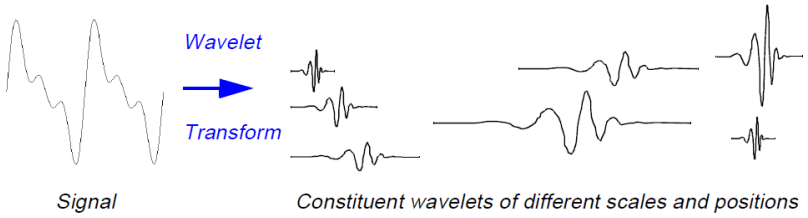


Fig 3.21 Wavelet Decomposition of a Signal

2. Discrete Wavelet Transform (DWT)
   The Discrete Wavelet Transform(DWT) is a wavelet analysis technique in which the scale and position of the wavelets are varied in powers of two which is called dyadic scales and positions. For calculation of DWT, series of filters are used which give the approximation and detail coefficients which are explained in the next section.

The wavelet transform method used in this project is DWT as the results can be more accurately determined by processing less number of data sets.

For most real world signal the low frequency content forms the most important part. For example consider human voice. If high frequency component is removed the voice sounds different but what is being said can still be understood. Now if low frequency components is removed then only gibberish is heard. Hence it can be said that the high frequency component imparts the flavor to the signal whereas the low frequency component forms the identity of the signal.

The DWT of a signal is calculated with the use of series of filters. The approximations (A) are the high-scale, low-frequency components of the signal whereas the details (D) are the low-scale, high-frequency components of the signal.

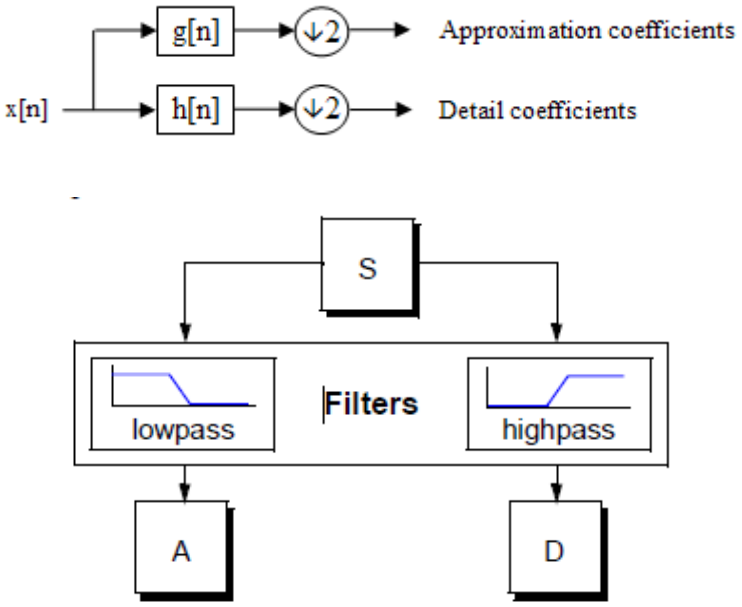
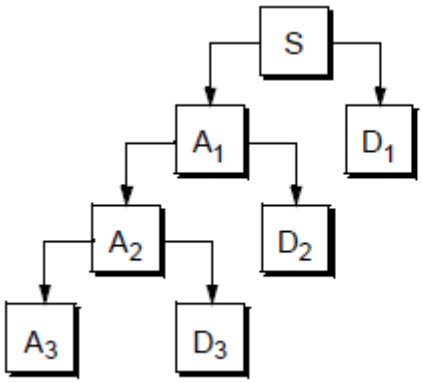


Fig 3.22 Wavelet decomposition into high and low frequencies

The original signal, S, passes through two complementary filters and emerges as two signals.
The decomposition process is iterated, with successive approximations being decomposed in turn. In this way the original signal is broken down into many lower-resolution components. This thus leads to a tree formation as shown in the figure below which is called the wavelet decomposition tree.



$$S = A_1 + D_1$$
$$= A_2 + D_2 + D_1$$
$$= A_3 + D_3 + D_2 + D_1$$

Fig 3.23 Wavelet Decomposition Tree

The detail coefficients (D) consist mainly of the high-frequency noise, while the approximation coefficients (A) contains much less noise than does the original signal. Hence in this way the decomposition coefficients of a signal are obtained.

Wavelet transforms also has another concept of the wavelet family and the choice of the mother wavelet is very important. There are various Wavelet Families from which the mother wavelet for analysis is chosen. Some of them are

1. Haar
2. Daubechies
3. Biorthogonal
4. Morlet

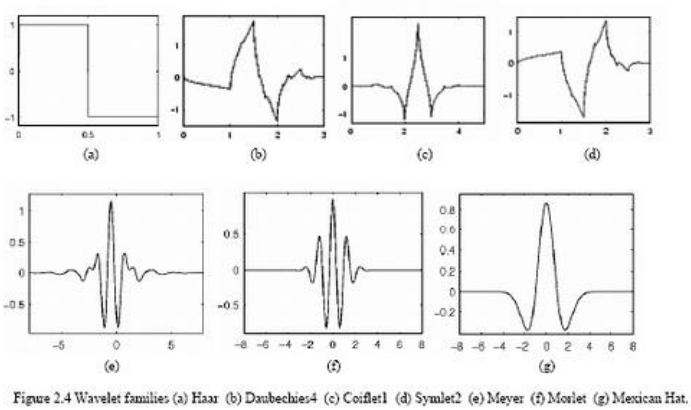


Figure 2.4 Wavelet families (a) Haar (b) Daubechies4 (c) Coiflet1 (d) Symlet2 (e) Meyer (f) Morlet (g) Mexican Hat.

Fig 3.24 Mother wavelets

When an ECG is recorded many kinds on unwanted noise is also recorded with it. These noises cause an alternate shift in baseline of the ECG signal. A process of removing the baseline drift of a signal is called as de-trending. Wavelet transform is used in this project to de-trend the ECG signal that is obtained from the sensor.

The Daubechies wavelet of the DWT wavelet family is selected because the shape of the ECG signal and that of db5 is same. Also Daubechies wavelet families are similar in shape to QRS complex and their energy spectrums are concentrated around lower frequencies. Finally, the signal is loaded into Wavelet Analysis and Design Toolbox available in Matlab and the following results were obtained.

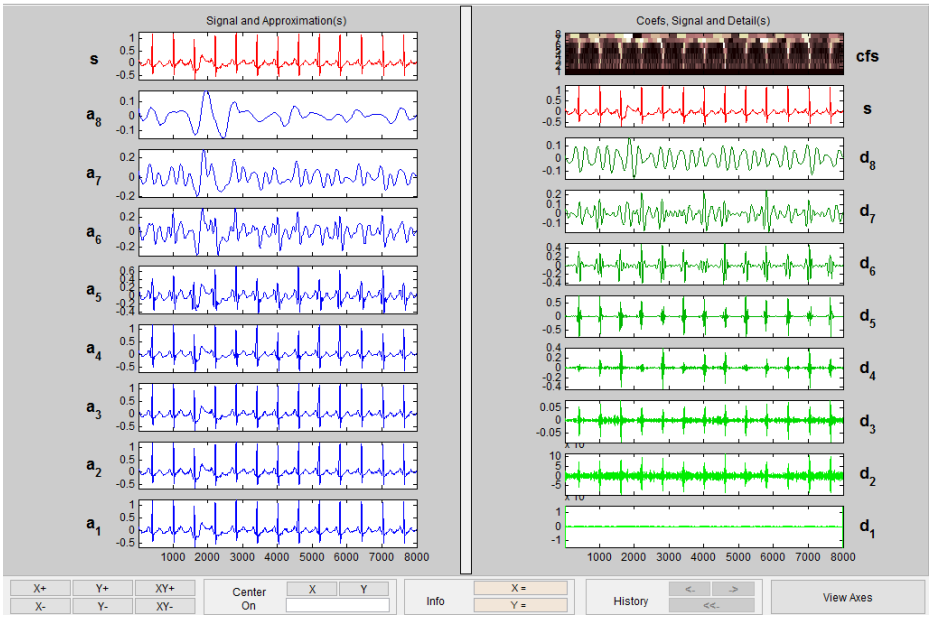


Fig 3.25 Wavelet Decomposition in different modes

Fig 3.25 shows the decomposition of the signal in separate mode. The right hand side depicts the high frequency decomposition components of the signal whereas the left hand side depicts the low frequency decomposition components of the original signal. The decomposition in tree mode can be seen in fig 3.26. It shows the wavelet tree on the left. The original signal and the last low frequency approximation of the signal is shown on the right. The complete decomposition of the signal with all high frequency components and the last approximation is seen in fig 3.27.

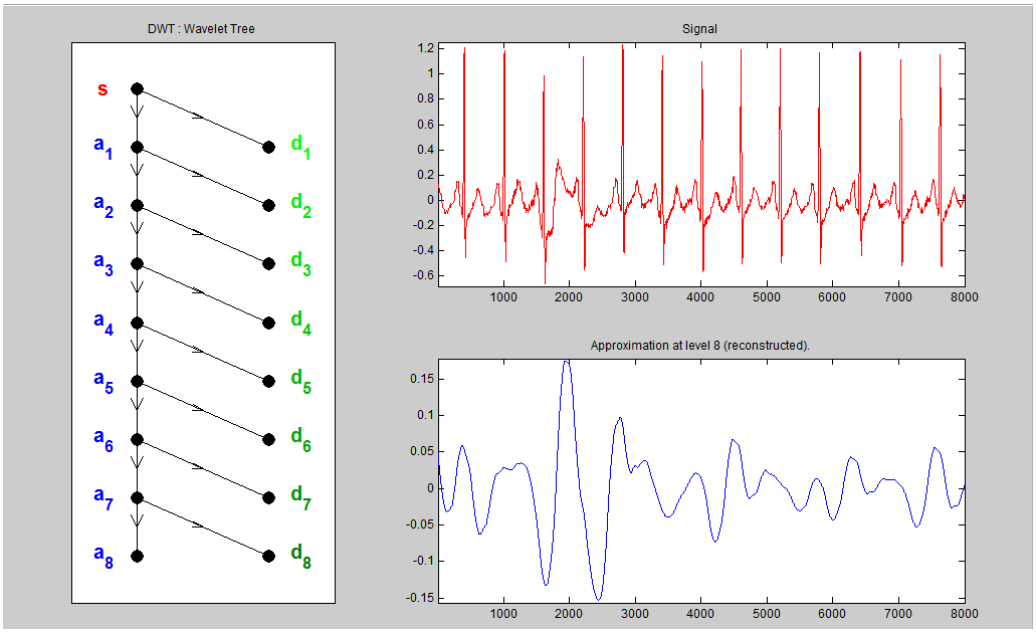


Fig 3.26 Decomposition in Tree mode

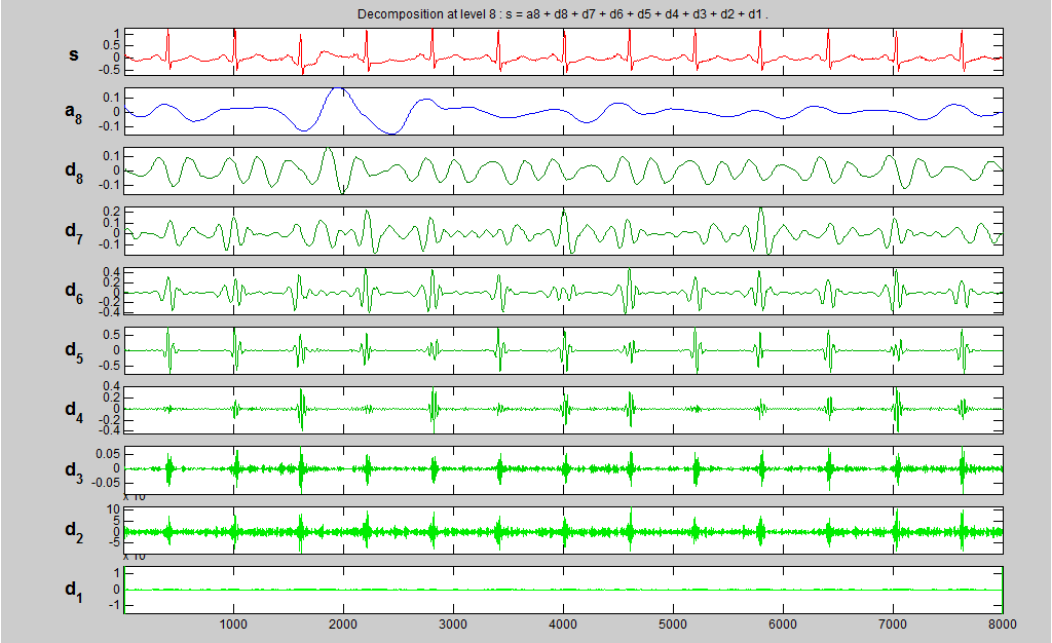


Fig 3.27 Complete Wavelet Decomposition

An overall stepwise approach to get complete wavelet decomposition to ECG signals is elaborated from here on. From the decomposition of ECG signal it is seen that the low frequency component is the cause for baseline wandering. Hence these components have to be removed from the original signal to get a clean signal which is free from baseline wandering.

From the above figures we can see that the low frequency component of the decomposed signal is A8.Therefore this component is subtracted from the original signal to get a de-trended signal.
De-trended Signal = Original Signal – (A8)

Thus in this way Wavelet Transform is used to remove the baseline wandering present in the ECG signal.

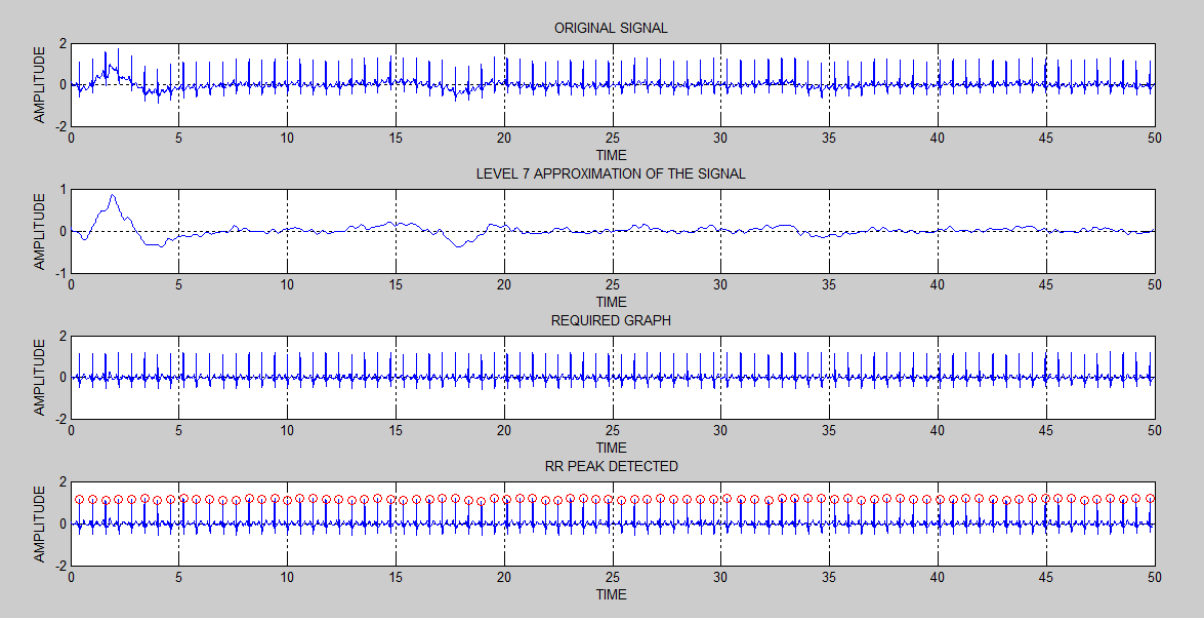


Fig 3.28 Removal of Baseline Wandering by Wavelet Decomposition and RR peak detection

The above figure shows the complete steps involved in removing the base line wandering of the signal using Wavelet transform and detecting the R-peaks. Part (a) of the figure shows the original ECG signal. Part (b) shows the level8 approximation obtained after Wavelet analysis on the ECG signal. The de-trended signal is shown in part (c). The RR-peaks is detected after detrending the signal as shown by part (d).

Another manner in which the results of the wavelet transformed signal was used, was to validate the results of the IIR filtered signals. The peaks of the IIR filtered signal were detected using threshold technique with the threshold set at seventy percent of the total ECG swing. The peaks of the wavelet decomposed ECG signal were also detected, using MATLAB's built in findpeaks() function. Since peak values obtained through IIR filtering and wavelet decomposition matched, each one served as a method to validate the results of the other.

3.3.3 Feature Extraction

Feature extraction is a method of extracting useful information from an otherwise meaningless ECG signal dataset. These features (ie. 'useful information') that are extracted from data samples are used by the classifer algorithms to learn how to categorize and classify samples belonging to different classes. Four types of features have been extracted from each of the data samples, and they are:

a. Time Domain features: This kind of feature extraction works on calculations made on the RR interval sequence. Seven time domain features have been extracted.

b. Non-Linear features: This kind of feature extraction also works on calculations made on the RR interval sequence. Here however, the mathematical equations that are used are obtained via graphical analysis. Three time domain features have been extracted.

c. Frequency Domain features: This kind of feature extraction applies Power Spectral Density (PSD) on different sections of the signal, to come up with features. Thirteen frequency domain features have been extracted.

d. Exogenous Input Auto Regressive (ARX) Model Coefficients: This kind of feature extraction involves breaking up the signal into two halves and using one half of the signal as input and other half of the signal as output to model a system. The coefficient that is calculated for the system can directly be used as features.

All the feature extraction operations performed make use of either the IIR filtered signal or the wavelet decomposed signal and not the original ECG. Time domain and non-linear features used IIR filtered signals, whereas the frequency domain and ARX modelled features used the wavelet transformed signal.

3.3.3.1 Time Domain

Time domain feature extraction is the simplest of the four types of features that have been extracted from the ECG signals. All calculations for time domain feature extraction have been performed on the IIR filtered signal, which was

described in section 3.3.2. In this type of feature extraction, the time instants of occurrences of R peaks is noted and the RR intervals (the time passed between two R peaks) is calculated. The series of RR intervals if plotted against time gives a visual representation of HRV of the ECG signal, and is called a tachogram. After obtaining the RR intervals, values as given below have been calculated and used as features:

a. Mean of RR intervals (RR_mean)
   This feature is the arithmetic mean calculated on the RR interval series and was calculated as follows

$$RR_{mean} = \frac{1}{n}\left(\sum_{i=0}^{n} RR_i\right) \qquad \text{(eqn 3.1)}$$

b. Standard Deviation of RR intervals (RR_std)
   This feature is the standard deviation of the RR interval series and was calculated using

$$RR_{std} = \sqrt{\frac{1}{n-1}\sum_{i=0}^{n}(RR_i - RR_{mean})^2} \qquad \text{(eqn 3.2)}$$

c. Mean Heart Rate (HR_mean)
   From the RR interval series, the average frequency of occurrence of the RR intervals per minute was calculated using eqn 3.3.3.1_3. The inverse was then taken which resulted in the mean heart rate.

$$HR_{mean} = \frac{60 \times 1000}{RR_{mean}} \qquad \text{(eqn 3.3)}$$

d. Standard Deviation of Heart Rate (HR_std)
   The standard deviation of the heart rate was calculated in a similar manner as the mean heart rate. First the standard deviation of the of the frequency of occurrence of the RR intervals per minute was calculated. Then, the inverse was taken to obtain the standard deviation of the heart rate.

$$HR_{std} = \frac{60 \times 1000}{RR_{std}} \qquad \text{(eqn 3.4)}$$

e. Root Mean Square of RR intervals (RR_rms)
   The square root of the mean of the sum of the squares of all the entries in the RR interval series results in the RMS of the RR interval series. The formula for the same is as follows:

$$RR_{rms} = \sqrt{\frac{1}{n-1}\sum_{i=0}^{n-1}(RR_{i+1} - RR_i)^2} \qquad \text{(eqn 3.5)}$$

f. Number of Intervals Varying by Larger than a Threshold (RR_50)
This feature is slightly different from the rest, in the sense that it involves an additional step of taking differences. While obtaining the RR interval series required taking successive differences of the time instants at which R peaks occurred, here, successive differences are taken for the values in the RR interval series itself. On this new series of difference, the number of time differences that are larger than 50ms are counted to yield the RR_50 feature.

g. Relative Number of Intervals Varying Larger than a Threshold (RR_r50)
The previous feature obtained divided by the total length of the the RR interval series gives rise to the final time domain feature. This can be represented by the following equation:

$$RR_{r50} = \frac{RR_{50}}{n-1} \times 100\% \qquad \text{(eqn 3.6)}$$

3.3.3.2 Non-Linear

Three non-linear features have been extracted for each of the samples in the dataset. There were two types of analysis done, one of which yielded two features while the other gave rise to a single feature. These two methods are given below:

i. Poincare Plot
This feature is extracted 'graphically'. A graph of Poincare points is plotted, where the horizontal axis value of all the points is some i-th value in the RR interval series, while the vertical axis value of the point is the (i+1)-th value of the RR interval series. Then, the standard deviations of the points along two different axes are calculated to yield two of three non-linear features denoted by SD1 and SD2 respectively. The axes along which the standard deviations are calculated are the x = y line and the x = -y line. A sample of the Poincare plot obtained for the first alcoholic sample in the dataset has been shown below.

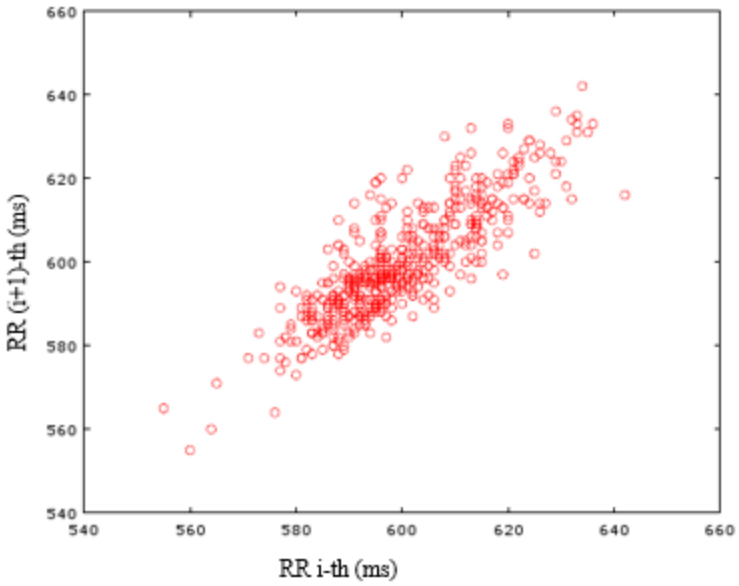


Fig 3.29 Poincare Plot

ii. Approximate Entropy

This feature is essentially a measure of how much irregularity exists within the RR interval series. The following steps explain how the feature value was calculated:

a. From the RR interval sequence, a set of $N + m - 1$ vectors of length m are formed. N is the total length of the RR interval series.
b. Then the maximum of the element-wise differences of all the pairs of vectors is found and stored.
c. Then a quantity C is calculated that counts the number of distance metrics that were lesser than a threshold r.
d. The natural logarithm of the quantity C is calculated
e. The sum of all such logarithmic values corresponding to all the vectors is obtained.
f. Similar steps are performed for vectors of length $m + 1$
g. The difference between the results for vectors of length m and $m + 1$ gives the approximate entropy.

Exact equations to implement the above steps are mentioned in [12]

3.3.3.3 Frequency Domain

The frequency domain features calculated are obtained from the power spectral density (PSD) of the ECG signal. In order to obtain good PSDs that accurately map to the power contained in the ECG signal, the signal needs to be free of as

much noise as possible. This is the reason wavelet transform is used to filter the signal while obtaining the frequency domain features.

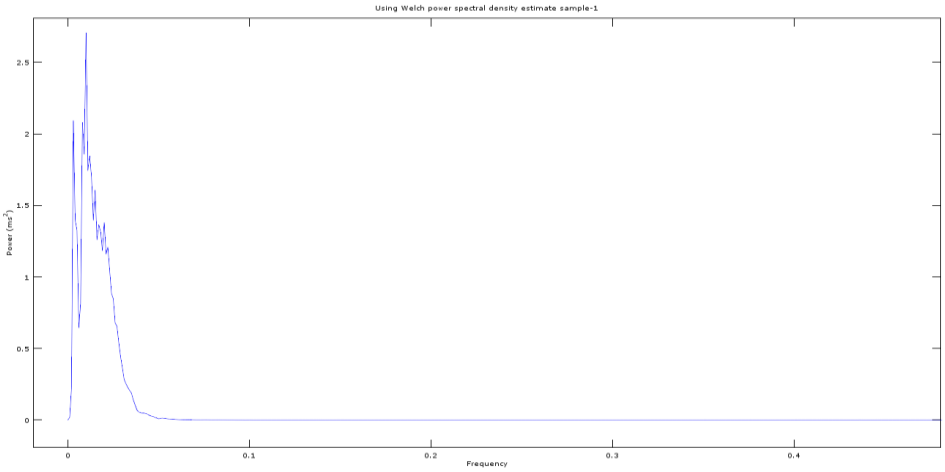


Fig 3.30 PSD of a sample from dataset used

Some features are calculated for the whole signal while the rest are calculated only within some frequency sections of the signal. The three main frequency ranges where features are calculated are the very low frequency (VLF) range, the low frequency (LF) range and the high frequency range (HF). VLF has a range from 0Hz to 0.04Hz, LF has a range from 0.04Hz to 0.15Hz, and HF is the range of all frequencies above 0.40Hz. The frequency domain features that have been extracted are:

a. Peak Frequency in VLF, LF, HF (pk_freq_vlf, pk_freq_lf, pk_freq_hf):

The amplitude of the peak frequency values in very low, low, and high frequency ranges of the PSD of the signal form the first three frequency domain features.

b. Absolute Power in VLF, LF, HF (ab_pow_vlf, ab_pow_lf, ab_pow_hf):

The total power contained in the very low, low and high frequency ranges of the PSD of the signal form the next three frequency domain features.

c. Total Power of the Signal (pw_ttl):

The seventh frequency feature used is the total power contained in the PSD of the signal.

d. Relative Power in VLF, LF, HF (rp_vlf, rp_lf, rp_hf):

The relative power of a certain band is calculated as the power in that band divided by the total power present in the signal. The folmula for VLF is given below (eqn 3.3.3.1_7). The formulae for LF and HF are similar.

$$rp_{vlf} = \frac{ab\_pow_{vlf}}{pw_{ttl}} \qquad \text{(eqn 3.7)}$$

e. Normalized Power in LF, HF (norm_lf, norm_hf):

The normalized power of the LF and HF range is the absolute power present in that range divided by the difference in the total power of the signal and the power contained in the VLF band. The equation for the normalized power in the LF range is given by (eqn 3.8) and that for the HF band is also similar.

$$norm_{lf} = \frac{ab\_pow_{lf}}{pw_{ttl} - ab\_pow_{vlf}} \qquad \text{(eqn 3.8)}$$

f. Ratio of Absolute Power of LF and absolute power of HF (ratio):

This last feature is obtained by taking the ration pf the power present in the LF range and that present in the HF range.

$$ratio = \frac{ab\_pow_{lf}}{ab\_pow_{hf}} \qquad \text{(eqn 3.9)}$$

3.3.3.4 Auto-Regressive Modelling with Exogenous Input (ARX)

The autoregressive (AR) model establishes a connection between the output variable and its previous values. It states that each value can be derived linearly from its previous values as well as a random Gaussian variable with mean zero, the noise. Using AR model, a signal sequence y(n) can be represented as:

$y(n) = a\_y(n-1) + a\_y(n-2) + a\_y(n-p) + \varepsilon(n)$ (eqn 3.10)
where $a\_(k = 1,2,\dots,p)$ are the model coefficients and ε(n) is a white noise series. In the condensed form it can be written as:

$$X_t = \sum_{i=1}^{p} a_i \, . X_{t-i} + \varepsilon_t \qquad \text{(eqn 3.11)}$$

where a_i are called the AR parameters

The AR model is further expanded to include ECG signal as input signal. The ECG signal obtained from the subject is divided into two halves .One half of the signal serves as input to the system and the other half serves as the output. The AR model now becomes an Auto-regressive model with exogenous (ARX) input. The representation of the signal is :

$X_t = \sum_{i=1}^{p} a_i \ . X_{t-i} + \sum_{i=1}^{q} b_i \ . Y_{t-i} + \varepsilon_t$ (eqn 3.12)

where Y_t is the input ECG signal

After the ECG signal is free from base line wandering, the signal is loaded into the System Identification Toolbox in MATLAB to calculate the ARX coefficient. The following steps are used to obtain ARX coefficients for one ECG sample.

1. The System Identification toolbox is opened from the list of apps available or by typing in the following command “systemIdentification” in the command window of Matlab.

2. In the System Identification app window, select Import data and then Time domain data.

3. The ECG data recorded is divided into two equal parts where the first half of the signal acts as input and the other half of the signal acts as the output.

4. Each of this ECG signal is first loaded into the Matlab workspace.

5. The variables representing the signal in the workspace are then entered into the System Identification toolbox. Also the sampling interval is specified.

6. From Estimate tab Polynomial model is selected.
The ARX model is already selected by default in the Structure list.|

7. The order field is edited to enter the range of poles, zeros and delays. na represents the number of poles , nb represents the number of zeros and nk represents the delay time which is he time taken by the input sample to affect the output sample.

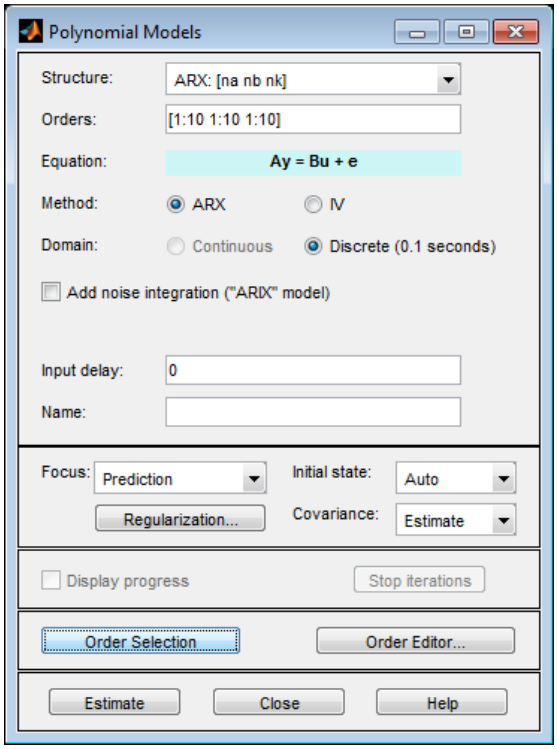


Fig 3.31 MATLAB Toolbox to select order of ARX polynomial

8. The "Estimate" button is clicked to open the ARX model Structure selection window which displays the model performance for each combination of model parameters.

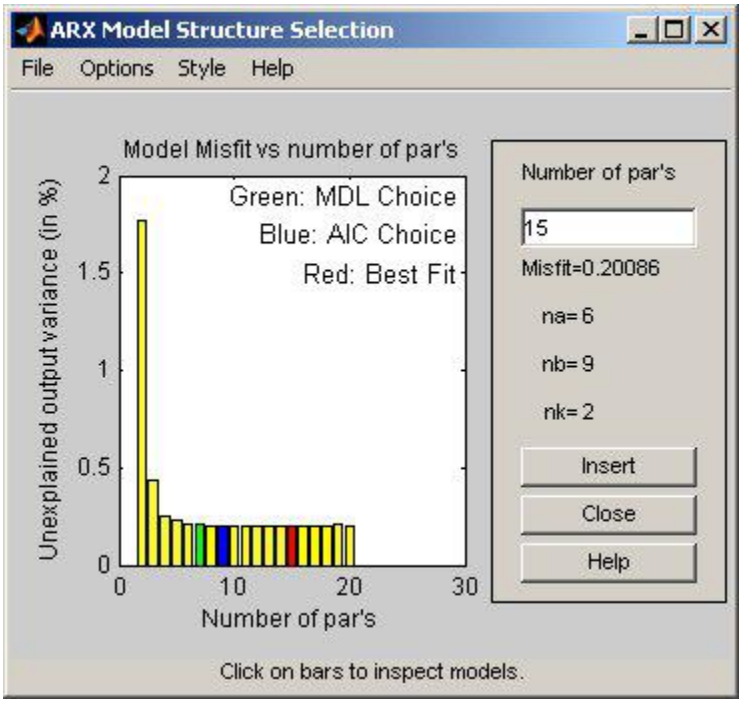


Fig 3.32 ARX model structure selection

9. A rectangle is selected which represents the optimum parameter combination. The parameters shown on this window are:

i. The horizontal axis is the total number of parameters which is the sum of number of poles and number of zeros

ii. The vertical axis, called Unexplained output variance (in %), is the portion of the output not explained by the model—the ARX model prediction error for the number of parameters shown on the horizontal axis.

iii. nk is the delay

Three rectangles are highlighted on the plot in green, blue, and red. Each color indicates a type of best-fit criterion, as follows:

a. Red - best fit minimizes the sum of the squares of the difference between the validation data output and the model output. This rectangle indicates the overall best fit.

b. Green - best fit minimizes Rissanen MDL criterion.

c. Blue - best fit minimizes Akaike AIC criterion

10. The “insert” button is then clicked which adds a new model board in the System Identification App

11. The selected model can then be viewed by double clicking the model from the model board.

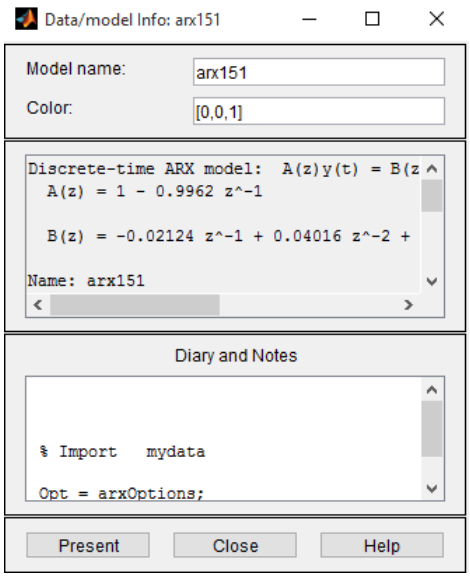


Fig 3.33 Coefficients of ARX model fit to required order and input/output signals provided

All the information related to the arx model is listed in this window. The model can also be seen on the command line of Matlab by clicking the "Present" button. In this way the ARX coefficients of the system is calculated. These ARX coefficients is used as feature for the classifiers.

In conclusion, seven time domain, three non-linear, 13 frequency domain and four or six ARX features (depending on the order of ARX polynomial coefficients) were obtained from each sample. These twenty six features are then fed into the classifiers to train them and obtain the optimal set of weights for classification of any new input data.

3.4 Classifiers

Classification is the process of identifying which sub category (or class) a certain sample belongs to. Several classification algorithms have been developed and tested for a number of datasets, for example, Support Vector Machines, Naive Bayes Classifier and Neural Networks. Some classifiers perform better than others for a certain application, or for a specific dataset. There are a number of parameters that are looked into which selecting a certain classifier for a certain application, like accuracy, training time, testing time, etc. Two classifier algorithms have been implemented. One of the classifiers is the Support Vector Machine (SVM), while the other is the Extreme Learning Machine (ELM). This section covers the basic concepts and ideas behind each of the algorithms used.

3.4.1 Support Vector Machine

A support vector machine (SVM) is a classifier that works on finding a decision boundary that can separate the classes of a dataset. Similar to a two dimensional problem where lines and polynomial curves are used to separate the data belonging to different classes, SVMs use separating surfaces/planes of higher dimensions called hyperplanes. SVMs have a primary hyperplane that behaves as the actual decision boundary for classification, but also uses two other hyperplanes in order to achieve the optimized primary hyperplane. Optimization

of the primary hyperplane is done generally solving a Lagrangian dual to the actual geometric equation that needs to be solved.

As the number of features that have been extracted for each samples in the dataset is twenty-six, all the data points and hyperplanes exist in the twenty-sixth dimension. This means that twenty-six coordinates exist to describe each data point. A hyperplane in such a dimension is given by (eqn 3.6.1_2), where w, x and b have 26 coordinates (eqn 3.6.1_1)

$$\begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_{26} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{26} \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_{26} \end{pmatrix} \qquad \text{(eqn 3.13)}$$

$$w^T x + b = 0 \qquad \text{(eqn 3.14)}$$

$$w^T x + b = 1 \qquad \text{(eqn 3.4.15)}$$

$$w^T x + b = -1 \qquad \text{(eqn 3.4.16)}$$

If (eqn 3.14) is considered to be primary hyperplane that takes on the role of being the decision boundary, there exist two other hyperplanes lying on either side of the primary hyperplane given by (eqn 3.15) and(eqn 3.14). The purpose of the two adjacent decision boundaries is to aid in arriving at a geometric optimization problem. What needs to be ensured is that the adjacent hyperplanes lie as far away from each other as possible without misclassifying any of the data samples. In other words, the margin between the primary hyperplane and the adjacent hyperplanes needs to be maximized. The margin/distance that exists between the adjacent hyperplanes can be given by (eqn 3.17), and this is the value that needs to be maximized.

$$margin = \frac{2}{||w||} \qquad \text{(eqn 3.17)}$$

$$min \ \frac{||w||^2}{2} \qquad \text{(eqn 3.18)}$$

Maximizing (eqn 3.17) is the same as minimizing (eqn 3.18). Equation (eqn 3.18) can be solved directly or, as per [19] the Lagrangian's dual can be solved. The Lagrangian dual for (eqn 3.17) is given by (eqn 3.19).

$$L(w, b, \alpha) = \frac{||w||^2}{2} - \sum_i^m \alpha_i \left[y^i (w^T x + b - 1) \right] \qquad \text{(eqn 3.19)}$$

Subject to constraints (as per Karush-Kuhn-Tucker);

$$\sum_{i=0}^m \alpha_i = 0 \qquad \text{(eqn 3.20)}$$

$$\sum_{i=0}^m \alpha_i y^i = 0 \qquad \text{(eqn 3.21)}$$

$$w = \sum_{i=0}^m \alpha_i y_i x_i \qquad \text{(eqn 3.22)}$$

These yields the final Lagrangian optimization equation:

$$max_\alpha W(\alpha) = \sum_{i=0}^m \alpha_i - \frac{1}{2} \sum_{i,j=0}^m y^i y^j \alpha_i \alpha_j K(x_i, x_j) \qquad \text{(eqn 3.23)}$$

In equation (eqn 3.23), m represents the total number of training samples supplied to the classifier, $x_{i/j}$ stands for the i-th or j-th training sample and $y_{i/j}$ stands for the class label of the i-th or j-th sample. α_i is the Lagrangian multiplier for the i-th sample. Lagrangian multiplier values are assigned based on whether they are support vectors or not, and K stands for the kernel function that is applied on the input dataset. More information on the application of and need for kernels is given later on in this section.

The equation for the decision boundary that is finally obtained is given by (eqn 3.24)

$$f(x) = \sum_{i=0}^{SVs} \alpha_i y^i K(x_i, x) + b \qquad \text{(eqn 3.24)}$$

where SVs stands for support vectors*.

Various algorithms exist to solve the Lagrangian dual. Here, a simplified version of the Sequential Minimal Optimization (SMO) algorithm proposed in [20] has been used.

Earlier, in equation (eqn 3.23) K was used to represent the use of a kernel on the dataset. Kernel provide ways of changing the dimension of a dataset using simple dot products, which remain computationally efficient.

---

* SVs = Support vectors are those points of the dataset through which the adjacent hyperplanes pass. The Lagrangian coefficient obtained for such data points is nonzero

By changing the dimension of the dataset the present feature space is mapped to another feature space. Using the right kernel an allows some data points that were not separable by hyperplanes in the original dimension to becoming separable in the new dimension that they are mapped to. Thus, this becomes a tremendously powerful method by which a larger set of data points can be classified correctly, helping increase the training accuracy of the algorithm. Some of the common kernels that are used are the polynomial kernel, Gaussian (also called the Radial Basis Function (RBF)) kernel and the wavelet kernel. Multiple kernels have
been used with the SVM classifier and based on the training accuracy obtained, the Gaussian kernel (given by eqn 3.25) has been selected.

$$K(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right) \qquad \text{(eqn 3.25)}$$

Another important concept about decision boundary type of classifiers like SVMs is regularization. It is common for a dataset to have a few outliers and anomalies. Since the algorithm attempts to classify all the samples correctly, sometimes the algorithm can overfit or go out of the way to include an anomalous sample to the class it had been assigned. As a result, the decision boundary that the algorithm comes up with does not remain generic to new test data that is fed in. Regularization is a method by which some equation parameter/coefficient can be tweaked and tuned to control the algorithm's sensitivity to outliers and anomalies. This control over how closely the algorithm should fit to the training data supplied to the algorithm is very important for real world applications, and has been implemented in the SVM classifier as the parameter 'C'. The use of a regularization parameter puts an additional constraint on the Lagrangian coefficients α. That is, apart from the constraints given by (eqn 3.20 to eqn 3.22), there is an additional constraint as given by (eqn 3.26)

$$0 < \alpha < C \qquad \text{(eqn 3.26)}$$

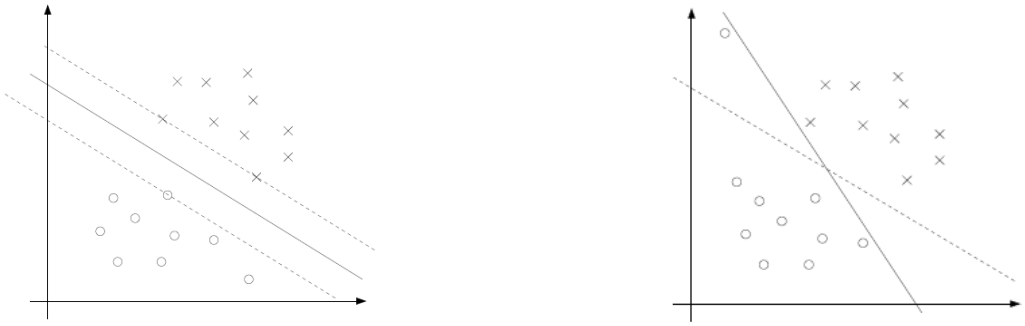


Fig 3.34 Importance of regularization

A final thought on SVMs is how the SVM, once trained with the weights of the decision boundary, is used to classify a new data sample. This process of classification of some i-th sample after obtaining a trained model is very straightforward. The (eqn 3.27) is calculated to obtain a number that represents which class the sample belongs to. If the number calculated is positive, then the sample belongs to the first class. In case the number is negative, it is classified to the other class.

$$pv = \sum_{i,j=0}^{SVs} y^i \alpha_i K(x_i', x_j') + b \qquad \text{(eqn 3.27)}$$

where pv is the prediction value and x_i' and x_j' is the test point needing to be classified into one of the two classes.

If the outcome of (eqn 3.27) is positive, the test sample is classified as belonging to one class, but if the outcome is negative, the test sample is classified to the other class.

$$class = \begin{cases} class\ 1 & if\ pv > 0 \\ class\ 2 & if\ pv < 0 \end{cases} \qquad \text{(eqn 3.28)}$$

To summarize all that was explained above and piece together how all the steps fall into place to train an SVM, the following steps have been provided:

1. The input matrix dataset is fed to the algorithm, and all the features are normalized to ensure that all the features contribute equally to the learning algorithm.
2. A combination of regularization parameter C and Gaussian variance value σ are chosen.
3. To this normalized data set, the desired kernel function is applied. If the kernel being used is a Gaussian kernel, then the σ chosen in the previous

step is used. If a different kernel is used that requires no parameter to be tuned, then only regularization parameter C is used.

4. Now, the SVM is actually 'trained' using some optimization algorithm like Sequential Minimal Optimization (SMO) to solve the Lagrangian dual and obtain a set of weights for the hyperplane. Further reference for the working of the SMO algorithm is provided in [20].
5. Once the weights for the hyperplane is obtained, the same kernel is applied to the normalized validation dataset which is sent to the trained classifier to calculate the accuracy.
6. Accuracy calculation may be performed either on a single validation dataset, or averaged over multiple folds of data using a technique like k-fold cross validation. For each fold the algorithm is retrained and the weights are recalculated, and validation is performed using the new weights
7. All the above steps are performed for new pairs of values of the regularization parameter C and Gaussian variance σ, and the pair that yields the best accuracy is chosen to train the final SVM model.

Once a trained SVM is present (ie. The weights for the hyperplane giving high training accuracy without being biased and without overfitting have been obtained), classification of a new sample is done as follows:

1. The sample to be classified is normalized using the same method that was used to normalize the original dataset
2. Features are then extracted from the normalized sample and fed to the classifier
3. The classifier then passes this sample through the Gaussian kernel and uses (eqn 3.4.1_16) to get a positive or negative number.

$$f(x_j) = \sum_{i=0}^{SVs} \alpha_i y^i K(x_i, x_j) + b \qquad \text{(eqn 3.29)}$$

4. If the expression on the right hand side of (eqn 3.29) is positive it belongs to a particular class, and if negative, it belongs to the other class.

As mentioned in the steps on how to train the SVM algorithm, the regularization parameter C and Gaussian kernel variance σ need to be fixed at the right values to obtain optimal accuracy. There is no real fixed or preferred method that is

used to fix these parameters. The method used to fix the parameters in this project has been described here.

1. Two arrays containing the values of C and σ are initialized.
2. For each value of C considered, a loop is run 200 (or 400) times
3. In each iteration of the loop, the training dataset is labelled into k-sets using the crossvalind() function.
4. One out of the 'k' folds is selected as the validation set, while the rest are taken as the training data.
5. The model is trained for such a dataset and validated using the validation samples as given by crossvalind().
6. Similar validation is performed for all of the k-folds to obtain the k-fold cross validation accuracy.
7. An array is maintained to keep track of which σ values yielded the most accuracy in each of the 200 iterations.
8. Similar steps are performed for all C values, and each array is appended to form a matrix.
9. The selection of C and σ that are picked the most in the 200 iteration loop are selected to be the values for which the final model is trained.

Crossvalind was found for 25 loops of crossvalind to average the randomness.

C/Sigma	0.01	0.03	0.1	0.3	1	3	10	30
0.01	0	2	1	12	170	6	4	5
0.03	0	2	1	11	171	8	4	3
0.1	0	1	1	12	177	2	3	4
0.3	0	2	1	6	186	3	1	1
1	0	5	0	7	131	47	8	2
3	0	7	2	13	26	126	16	10
10	0	11	5	30	12	0	114	28
30	0	9	3	21	20	0	123	24

Fig 3.4.1_2 A count of the C and sigma used for different loops in the case of ARX order 5 features is given. It is clear that C = 0.3 and sigma = 1 is chosen most.

3.4.2 Extreme Learning Machine [ELM]

A neural network is a processing unit consisting of sub units called neurons, which are interconnected to each other. These interconnects are assigned weights, representing the acquired knowledge, which may or may not be changed as the classifier is trained using the training samples. A neural network consists of an input and output layer, along with one or more hidden layers.

In machine learning, if the classifier is being trained using labelled data, that is, the corresponding target output is given for a certain input, it is known as supervised learning. If the classifier is being trained using unlabeled data, and clustering algorithms are required, it is known as unsupervised learning.

A feed forward network is one in which the values at the input are propagated towards the output through the hidden layers without being looped back to any preceding layers as an input. Back propagation consists of values that are fed forward through to the output, the error is calculated, and fed back to the preceding layers in order to correct the weights.

ELM is a single hidden layer, feed forward neural network. The weights connecting the input nodes to the hidden layer nodes are assigned randomly and never updated. The weights between the hidden nodes and output nodes are learnt in a single step. These models can learn several times faster than neural networks trained using the back-propagation algorithm. They are also known to produce decent generalization capability.

The architecture of an ELM model is fixed a-priori, that is, the number of neurons in each of the layers is fixed before training. The number of input layer nodes is equal to the size of the feature set being input to the classifier. The number of output neurons is equal to the number of classes of samples being fed. The number of hidden layer neurons is chosen based on which value gives highest accuracy without over-learning.

Kernels are a computationally efficient way of changing the dimensions of the feature set of a sample using dot products. If a kernel is used before the classifier, the dimensions of the input layer will change to the size of feature set after kernel is applied. Examples of some popular kernels include Radial Basis Function [RBF] Kernel, Polynomial Kernel, Fisher Kernel, et cetera. The kernel in use in this project is the RBF kernel. It is given by (eqn 3.30) where x is the feature set of a training sample, $\boldsymbol{\mu}$ is one of the kernel centres, and σ is the distance metric that is varied to check which gives best accuracy.

$$f(x,\mu) = \mathrm{e}^{-\left(\frac{(\mathrm{x}-\mu)^2}{2\sigma^2}\right)} \qquad \text{(eqn 3.30)}$$

The dataset is split into training data and testing data. The training data is used to train the classifier, after which the test data is used to test the accuracy of the classifier. The percentage of training and testing data is varied in order to prevent under-learning and over-learning and obtain best accuracy.

The Input Weights and Biases of Hidden Neurons are generated and assigned randomly. These values are never changed. Let the Input Weights be designated as W and the Bias as B. The dimensions of W is (n,k), where n is the number of hidden layer neurons and k is the dimension of the feature set of sample after passing it through a kernel. The dimensions of B is (n,p), where p is the number of input training samples given as a batch. If so, each column vector of matrix B will be identical.

$$\mathrm{W} = \begin{bmatrix} w_{11} & \cdots & w_{1\mathrm{p}} \\ \vdots & \ddots & \vdots \\ w_{\mathrm{n}1} & \cdots & w_{\mathrm{np}} \end{bmatrix} \qquad \mathrm{X} = \begin{bmatrix} x_{11} \\ \vdots \\ x_{\mathrm{p}1} \end{bmatrix} \qquad \mathrm{B} = \begin{bmatrix} b_{11} \\ \vdots \\ b_{\mathrm{n}1} \end{bmatrix}$$

The Input Weights are multiplied with the input training data and the Bias is added to it. Let the training data vector be designated as X. The output here is designated as H_{temp}, given in (eqn 3.31).

$$H_{\mathrm{temp}} = W \times X_{\mathrm{train}} + B \qquad \text{(eqn 3.31)}$$

The Activation Function is used to calculate the output of each neuron in the hidden layer. Examples of Activation functions include Sigmoid, Sinusoidal, Hard Limit, Triangular Basis function, Radial Basis functions. H_{temp} is the input argument to the activation function, which results in H, as shown in (eqn 3.32)

$$H = ActivationFunction(H_{\mathrm{temp}}) \qquad \text{(eqn 3.32)}$$

The activation function that is used in this project is the Sigmoid function, given by (eqn 3.33).

$$\mathrm{H} = \frac{1}{1+e^{-H_{\mathrm{temp}}}} \qquad \text{(eqn 3.33)}$$

It is to be noted that H has the same dimensions as $H_{temp.}$

The Output Weights, denoted by W_o, are calculated by multiplying the Moore-Penrose pseudo-inverse of H with the targets of training data, denoted by T, whose elements represent the target values of each class of training data. This is represented in (eqn 3.34)

$$T = \begin{bmatrix} t_{11} \\ \vdots \\ t_{c1} \end{bmatrix} \quad W_o = (H^T)^+ T^T \quad \text{(eqn 3.34)}$$

Once the classifier is trained, that is, all weights and biases are obtained, it is tested with the test sample data set.

First, the Input Weights, W, are multiplied with the Input Feature dataset (after passing through kernel function with centres as the same as that of training phase), given by $X_{test,}$ and added with the Bias, B, to obtain H_{test} as given by the (eqn 3.35).

$$H_{\text{test}} = W \times X_{\text{test}} + B \qquad \text{(eqn 3.35)}$$

The output of activation function is calculated by passing Htest as an argument to the activation function that was used in the training phase as well.

$$H = ActivationFunction(H_{\text{test}}) \qquad \text{(eqn 3.36)}$$

The actual output of testing data, Y is obtained by multiplying the output of activation function, H, with the Output Weights, W_o, which was assigned during the training phase. This is represented by (eqn 3.37).

$$Y = H^T W_o \qquad \text{(eqn 3.37)}$$

The predicted output class is the index of the maximum value in the output vector Y, as given by the (eqn 3.38).

$$Predicted\ Output = index(\max(Y)) \qquad \text{(eqn 3.38)}$$

ELM has been found to be good at learning easy functions and performing well for small number of labelled data. They are also incredibly fast to train and have fewer parameters that need to be trained. They provide comparable accuracies to that of other classifiers like Support Vector Machines [SVM], and other deep

neural network architectures, for a highly reduced training period and reduction in size of the neural network, owing to the single hidden layer and random initial assignment of input weights, which are not modified during the rest of the training process.

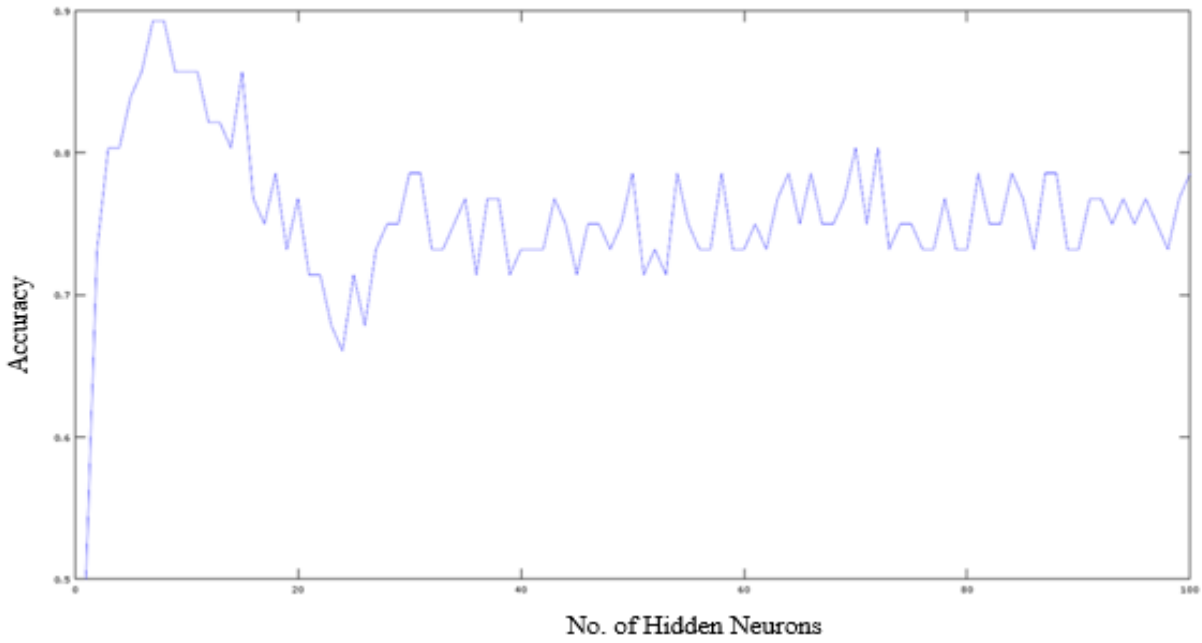


Fig 3.35 Accuracy for a range of hidden neurons

The manner in which the optimal value of hidden neurons and were obtained was by running a loop of combinations of sigma and hidden neurons multiple times to get all combinations of the two parameters to train over the dataset multiple times. The number of hidden neurons and which produced the largest accuracy in the loop were used to develop the model. In each of the loops, a track of accuracy and hidden neurons was maintained. This helped especially to see that the algorithm's accuracy peaked in the one to thirty neurons range, but with further increase in neurons, the algorithms would begin to overfit and accuracy would fall. The plot for one such iteration is given in Fig 3.35

3.4.3 Validation

Validation is an incredibly important part of testing how well a classifier works. Having trained the classifier for a certain portion of the dataset, it is essential to find out how well the classifier performs when exposed to a set of test data it has not seen before. Sometimes, the data that a classifier is trained for might result in the classifier being biased towards a certain class and might perform poorly when given test data that belongs to a different class. So it is important to

test the classifier using different test data sets or combinations of data for a classifier which has been trained for a corresponding different combination of training data set, so that upon averaging out the accuracies obtained from each train-test combination, we obtain an overall generalised picture of how the classifier performs in the real world and allows us to prevent over-fitting of the classifier. In cross-validation, the entire dataset is split into subsets of training and testing data, which are complementary to each other, and several rounds of training and testing are done, and results averaged over all the rounds of validation.

There are two types of cross validation, namely Exhaustive Validation and Non-Exhaustive Validation. Exhaustive cross-validation consists of Leave-P-Out Cross-Validation and Leave-One-Out Cross-Validation. In exhaustive cross validation, all possible combination of samples of data set are chosen and used to train and test the classifier. Non-Exhaustive cross-validation consists of K-Fold Cross-Validation, Hold-Out Method and Repeated Random Sub-Sampling Validation. Non-exhaustive cross validation methods do not include all possible combinations of samples from the dataset for training and testing the classifier. These methods are an approximation of the leave-p-out cross validation method that falls under exhaustive cross-validation.

3.4.3.1 Leave-One-Out Cross-Validation

This is a type of exhaustive cross-validation and an explicit example of leave-p-out cross-validation for p=1, in which one sample from the dataset is chosen for testing and all other samples are chosen to train the classifier per iteration, and this procedure is iterated for as many times as there are samples in the dataset, only that the sample chosen for testing is different every single iteration. The accuracies over every single iteration are averaged over and an average accuracy is obtained. This method is better than leave-p-out cross-validation in that the number of combinations of train-test data are numerically equal to the number of samples in the dataset. For example, in a dataset having 100 samples, in each iteration, one sample is chosen as the test data while the other 99 samples are chosen as the training data.

3.4.3.2 K-Fold Cross-Validation

This is a type of non-exhaustive cross-validation in which the dataset is randomly partitioned into k number of equally sized subsets, and in each of k number of iterations, one of the subset is chosen as the test data set while the rest of the k-1 subsets constitute the training data set, and the subset chosen to test and the subsets chosen to train the classifier are unique in every iteration of training and testing. The accuracies obtained in each k^{th} "fold" are averaged out to obtain an average accuracy that gives us an idea about the performance of the classifier. The advantage of this method of cross validation is that every single subset is used for both training and testing and a certain subset is used for testing only one time. The value of k determines the percentage of the dataset that is used for training and testing. For example, if k=4, for a dataset having 100 samples, the dataset is split into four subsets of 25 samples each. In each iteration, 75 samples are used for training and 25 samples are chosen as the test dataset.

This procedure happens three more times for the other three subsets of 25 samples, while the rest of the 75 samples in each fold participate as the training set. The accuracies over the four folds are averaged and presented as the accuracy of the classifier. When k equals the number of samples in the dataset, it becomes nothing but leave-one-out cross-validation.

3.4.3.3 Confusion Matrix

A confusion matrix is a table of values which provides us with a way to visualise the performance of an algorithm, or in the case of pattern classification problems, the performance of a classifier. The rows represent instances of the actual class while the columns represent instances of the predicted class. From this matrix, we can get to know how many times the classifier is classifying the sample correctly and how many times a certain class of sample is misclassified as another class, and the number for each class. The table of confusion (also called the confusion matrix) is one consisting of two rows and two columns. The four elements of this matrix give us the number of True Positives, False Negatives, False Positives and True Negatives. True positives are those samples which were classified correctly for a specific class. False negatives are those in which the actual label was that of the class we are calculating the matrix for, and the classifier predicted wrongly. False positives are those in which the predicted label was that of the class for which we were finding the matrix for,

but the actual label was that of another class. True negatives are those in which all the other classes apart from the class we were calculating the matrix for, were classified correctly.

Total Population	Prediction Positive	Prediction Negative
Condition Positive	True Positive	False Negative
Condition Negative	False Positive	True Negative

Table 3.2 Confusion Matrix

Sensitivity is a measure of the proportion of positives that are correctly classified as positives. It is also known as recall, hit rate or true positive rate.

$$Sensitivity = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad \text{(eqn 3.39)}$$

Specificity is a measure of the proportion of negatives that are correctly classified as negatives. It is also known as true negative rate.

$$Specificity = \frac{True\ Negative}{True\ Negative + False\ Positive} \quad \text{(eqn 3.40)}$$

Both of these are measures of the performance of a classifier for a certain dataset.

The significance of calculating the confusion matrix and from that, the sensitivity and specificity is that it allows us to find biases in the system if there exist any. For example, if it seen that the sensitivity of a system is low but the specificity is high, it would mean that the decision hyperplane (in the case of SVM) or the neural connections (in the case of ELM) might be biased toward the negative class. Getting to know such information would help tweak parameters of the system to correct it from being biased toward the negative class.

3.5 REAL TIME APPLICATION:

As the title suggest “A real time application to classify alcoholics from ECG Signals” , the ECG signal was captured in real time using the AD8232 heart rate monitor and Raspberry pi, features were extracted and the signal was classified.

The issue of absence of an analog pin on Raspberry pi was removed by the use of an ADC which is the MACP3008. The problem of mismatch of the sampling rate of the ADC and the Raspberry Pi was removed by the use of time library in Python which ensured that the Raspberry Pi read values every 1millisecond. The below image shows the ECG signal in real time on the CRO.

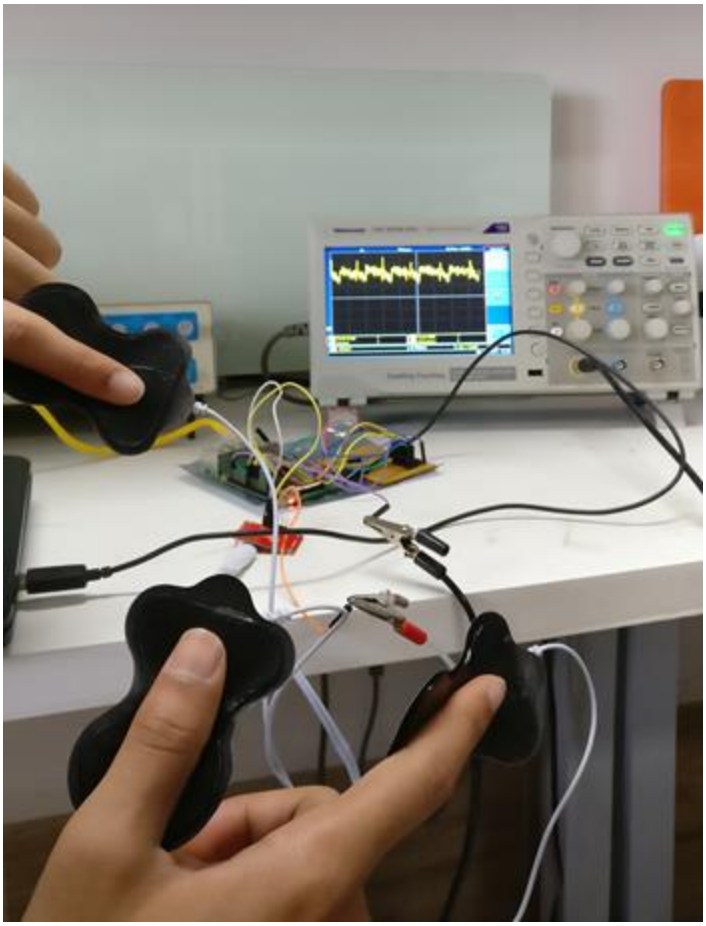


Fig 3.36 Real time capture of ECG signal

A Graphical User Interface (GUI) has also been created in python for easy demonstration of the real time application built here. The module used for GUI creation in python is know as Tkinter. All the above mentioned processing part of feature extraction and classification was implemented on Raspberry Pi.

3.5.1 Tkinter:

Tkinter is the Python's application programming interface(API) for Tk GUI toolkit library. Tk is a free and open source library used for creating a GUI across many programming languages.

3.5.2 Graphical User Interface (GUI)

The image below shows a GUI that is built for this project.

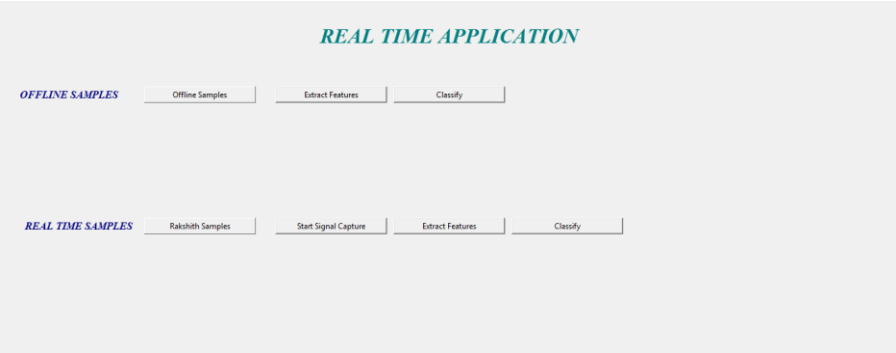


Fig 3.37 GUI

The first row of the GUI is to implement the offline samples available. It has buttons to load the signal, extract features and also to classify it based on the features extracted.
The second row of the GUI to implement a real time process wherein the user will click the “Start Signal Capture” and the signal will be captured for a given interval of time. On clicking “Extract features” button the features will be extracted from the signal and will be further used for classification.
All the required codes for signal capture, feature extraction and classifier are written in Python so that it can be easily integrated with GUI which is also written in Python.

The snapshots below show the entire process which is Signal capture, feature extraction and classification:

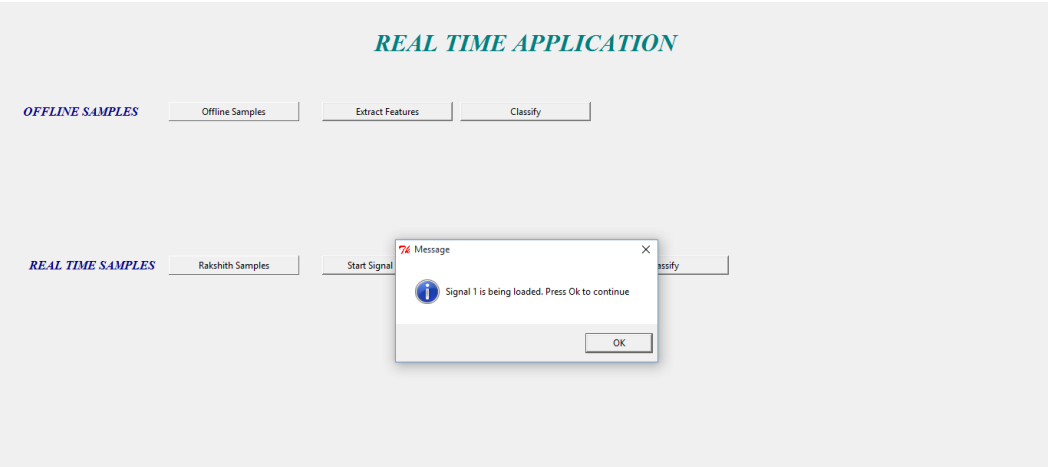


Fig 3.38 ECG signal loading process is started

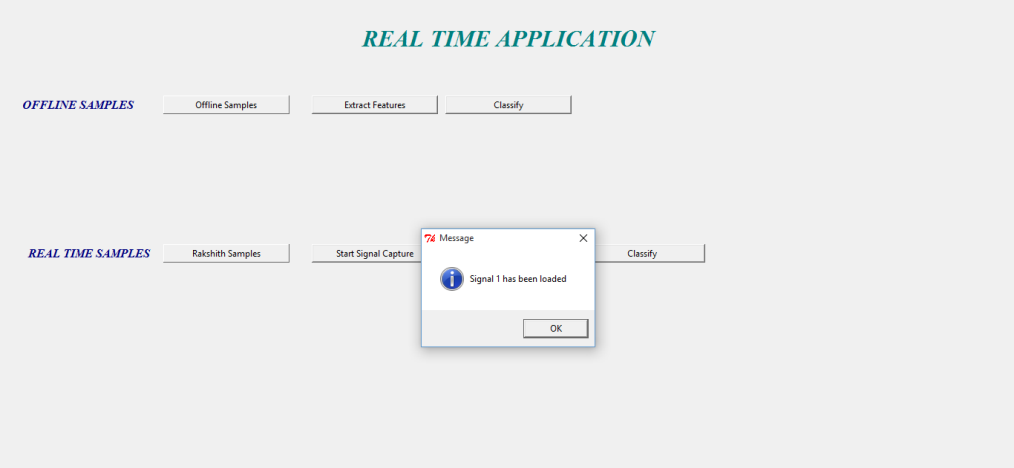


Fig 3.39 ECG signal is loaded

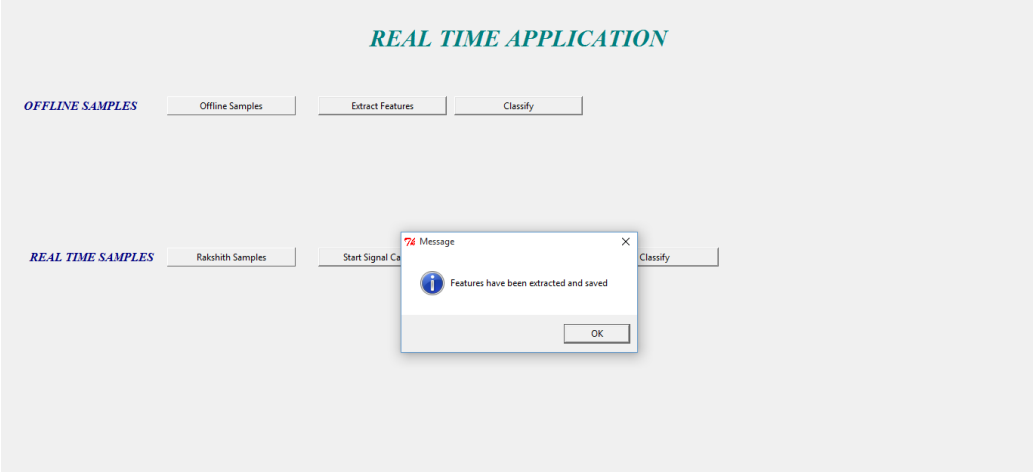


Fig 3.40 Feature extraction completed

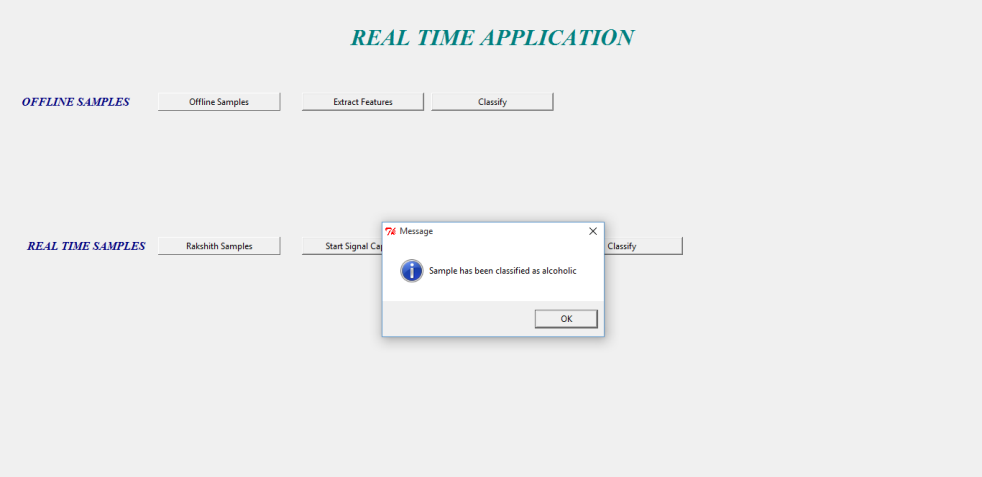


Fig 3.41 Classification completed

CHAPTER – 4

RESULTS AND DISCUSSION

This chapter is dedicated to describing the results obtained by the analyses performed in the project. For all the results, 28 alcoholic and 28 normative samples were used to train the system. Different features sets were used to train the classifier. A novel set of features that was used

4.1 Results of Pre-processing

Applying eight step wavelet decomposition on the ECG signal proved fruitful in separating the baseline wandering component from the signal. The eight level of decomposition when subtracted from the original signal resulted in a flat base ECG signal whose peaks could be detected accurately and PSD could be calculated without ambiguity.

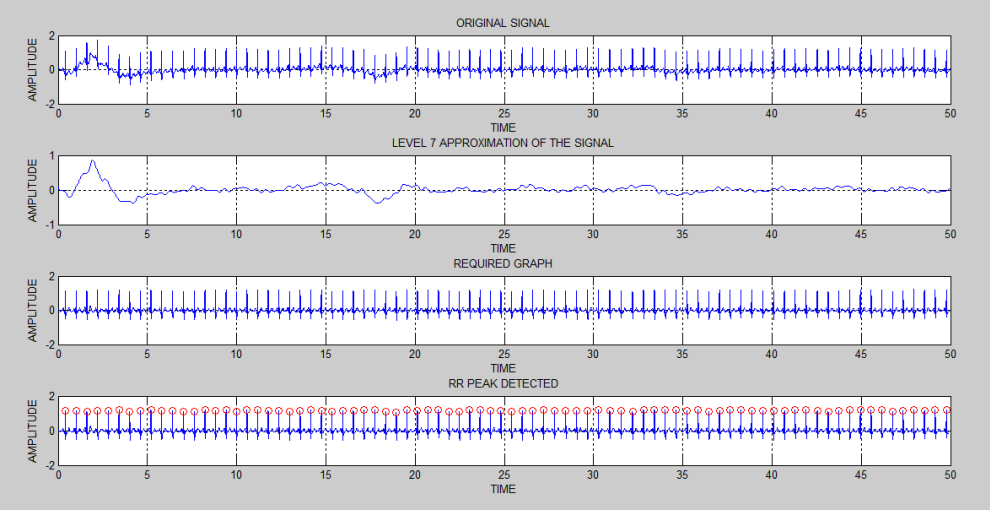


Fig4.1 Removal of Baseline Wandering by Wavelet Decomposition and RR peak detection

The above figure shows the complete steps involved in removing the base line wandering of the signal using Wavelet transform and detecting the R-peaks. Part (a) of the figure shows the original ECG signal. Part (b) shows the level8 approximation obtained after Wavelet analysis on the ECG signal. The de-trended signal is shown in part (c). The RR-peaks is detected after de-trending the signal as shown by part (d).

4.2 Results of Feature Extraction

Seven time domain, three non-linear, thirteen frequency domain and four ARX features were extracted from each of the fifty six ECG signals used to train the system. The array below gives a consolidated list of all the features that were extracted:

[RR_mean,	RR_std,	HR_mean,	HR_std,
RR_rms,	RR_50,	RR_r50,	sd1, sd2,
ApEn,	pk_freq_vlf,	pk_freq_lf,	pk_freq_hf,
ab_pow_vlf,	ab_pow_lf,	ab_pow_hf,	pw_ttl,
rp_vlf,	rp_lf,	rp_hf,	norm_lf,
norm_hf,	ratio,	ARX_coeff1,	ARX_coeff2,
ARX_coeff3,	ARX_coeff4,	ARX_coeff5,	ARX_coeff6]

4.3 Results of SVM

The process of selecting the C and σ pair, training the SVM, obtaining the k-fold validation accuracy have been mentioned in Chapter 3. Here, the results obtained for specific groups of feature sets used and the corresponding confusion matrices has been provided.

The accuracies and optimal C and σ pair obtained for the different types of feature sets are as given the table below.

Features Used	Optimal (C, σ) Pair	8-Fold Accuracy
Time, non-linear, frequency	(0.1, 0.3)	80%
Time, non-linear, frequency, ARX order 3	(0.3, 1)	82%
Time, non-linear, frequency, ARX order 5	(0.3, 1)	86%

Table 4.1 Results of SVM for different feature sets

From the table above, it is seen clearly that the accuracy of the system improves when ARX coefficients are used along with the time domain, non-linear and frequency domain features.

It is also important to understand how many samples of each class were classified correctly. For this the confusion matrix was obtained and the sensitivity and specificity (eqn 3.4.3.3_1 and eqn 3.4.3.3_2) were calculated for each of the cases given above in (Table 4.2).

Features Used	Confusion Matrix	(Sensitivity, Specificity)
Time, non-linear, frequency, ARX order 5	25 3 5 23	(89%, 82%)

Table 4.2 Confusion matrix for SVM with ARX features

The table above shows that the system is very slightly biased toward the positive class since a few more samples in the negative class are being misclassified. However, this is not of much concern as both sensitivity and specificity are well above 80% accuracy.

4.4 Results of ELM

The process of selecting the number of neurons in the hidden layer and σ, training the ELM, obtaining the k-fold validation accuracy have been mentioned in Chapter 3. Here, the results obtained for specific groups of feature sets used and the corresponding confusion matrices has been provided.

The accuracies and optimal number of hidden layer neurons and σ obtained for the different types of feature sets are as given the table below.

Features Used	Optimal (Number of Hidden Layer Neurons, σ)	7-Fold Accuracy
Time, non-linear,	(6, 1.4)	89.29%

frequency		
Time, non-linear, frequency, ARX order 3	(7, 2)	92.86%
Time, non-linear, frequency, ARX order 5	(15, 1.8)	94.64%

Table 4.3 Accuracies of ELM for different feature sets

Features Used	Optimal (Number of Hidden Layer Neurons, σ)	Leave-One-Out Validation Accuracy
Time, non-linear, frequency, ARX order 5	(6, 2)	92.86%

Table 4.4 Accuracy of ELM for Leave One Validation with ARX order 5

From the table above, it is seen clearly that the accuracy of the system improves when ARX coefficients are used along with the time domain, non-linear and frequency domain features.

It is also important to understand how many samples of each class were classified correctly. For this the confusion matrix was obtained and the sensitivity and specificity (eqn 3.4.3.3_1 and eqn 3.4.3.3_2) were calculated for each of the cases given above in (Table 4.5).

Features Used	Confusion Matrix	(Sensitivity, Specificity)
Time, non-linear, frequency, ARX order 5	26 2 4 24	(92.86%, 85.71%)

Table 4.5 Confusion matrix for ELM with ARX features order 5

The time above shows that the system is very slightly biased toward the positive class since a few more samples in the negative class are being misclassified. However, this is not of much concern as both sensitivity and specificity are well above 85% accuracy.

4.3 Comparative Results and Points of Discussion

Accuracies obtained for k-fold cross validation for SVM and ELM are compared in Table 4.6 and the sensitivity and specificity obtained for SVM and ELM are given in Table 4.7.

	SVM	ELM
Without ARX Coefficients	80%	89%
With ARX Coefficients of order 5	86%	94%

Table 4.6 Comparative results of SVM and ELM

	SVM	ELM
Sensitivity	89%	92.86%
Specificity	82%	85.71%

Table 4.7 Comparative Sensitivity and Specificity of SVM and ELM

Points of Discussion

The main points that can be taken out of this projects include the fact that HRV analysis on ECG signals provides a valid method to extract features that can be used by classifiers to detect chronic alcoholics from non-alcholics with high cross validation accuracy. Along with this, it can be seen that the neural network kind of ELM algorithm performed better than the SVM in all situations (training without ARX features and with ARX features). One of the crucial takeaways from the project is the use of ARX modelling of two sections of an ECG signal to provide useful features to classifiers. The coefficients of the ARX model generalized well for both algorithms, and a substantial increase in accuracy was seen for both.

CONCLUSION AND FUTURE WORK

Summary:

There was a lot of learning and knowledge gained by the end of this project. First, the ECG signal and its capture by building a hardware circuit was explored and understood. Then, study of papers led to exposure toward the physiological effects of alcohol on the body and the change in heart rate variability due to alcoholic consumption. After this, various techniques to extract features were explored and implemented. Feature extraction also required reading up on FIR filter, IIR filters and Wavelet Transforms. This was followed with exploring classifiers like the Support Vector Machine and Extreme Learning Machine which were to be applied to the features extracted to classify samples. Finally, signal sampling, ADC interface and sensor interface to the Raspberry Pi was implemented to create a Real Time system with a GUI.

At the end of this project it could be seen that meaningful features could be extracted from HRV analysis applied to the ECG signals, and that these features provided good accuracies for both classifiers. It was also seen that ELM performed better than SVM by around five percent of training accuracy. Another crucial point that could be drawn from this project is the use of ARX coefficient as features that improved the accuracy of both the algorithms substantially.

Future Work:

The real time application developed was tested for signals recorded for normative subjects only. To verify the complete working of the real time system, the ECG signal of an alcoholic subject needs to be obtained and classified. Currently, the ELM algorithm is optimized by 'brute force' through multiple trial and error loops. The plan ahead, is to prevent this randomness and utilize a technique called Meta-Cognitive Learning to ELM to make it learn more intelligently and accurately the first time around itself. Another point for future work to further improve the accuracy of the classifiers, is to use non-linear ARX model coefficients. The final goal of the project is to be able to come up with an algorithm that is able to classify subjects into multiple classes based on the level of alcohol intake.

REFERENCES

[1] Ajay Bharadwaj and Umanath Kamath, Cypress Semiconductor Corp., "Techniques for accurate ECG signal processing" [Online] Available: http://www.eetimes.com/document.asp?doc_id=1278571

[2] NTHU, "ECG Circuits, Signal Sampling and Digitalization" [Online] Available: https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0ahUKEwins5jwg43TAhWCq48KHcxPAysQFgggMAA&url=http%3A%2F%2Flms.nthu.edu.tw%2Fsys%2Fread_attach.php%3Fid%3D692351&usg=AFQjCNFJqOwwhxZZG0FP1yQNMuNpHBV73Q&sig2=e_-aLgXrLbFMIB-XhTd2Kg

[3] Lv Jinhua and Xu Yanyi, "Circuit Design for Front-End Electrocardiograph", International Journal of Multimedia and Ubiquitous Engineering Vol.11, No.5 (2016), pp.345-354

[4] Pico Technoology, "Electrocardiogram (ECG) circuit for use with oscilloscopes". [Online] Available: https://www.picotech.com/library/application-note/electrocardiogram-ecg-circuit-for-use-with-oscilloscopes

[5] Ping Shi, Ying Chen, Ming-Ming Guo and Hong-Liu Yu, "Acute Effects Of Alcohol On Heart Rate Variability: Time-Related Changes And Gender Difference", Biomedical Engineering: Applications, Basis and Communications,Vol. 26, No. 3 (2014) 1450048 (10 pages)

[6] U. Rajendra Acharya, K. Paul Joseph, N. Kannathal, Choo Min Lim, Jasjit S. Suri, "Heart rate variability: a review", Med Bio Eng Comput (2006) 44:1031–1051

[7] William Evans, "The Electrocardiogram of Alcoholic Cardiomyopathy", British Heart Journal, 21(4), (Oct. 1959): pp.445-456

[8] Kusuma Ramanna1, Fazal M Gahlot2, Nagaraja Puranik1, "Electrocardiogram changes and heart rate variability during moderate exercise

in chronic alcoholics", International Journal of Medical Science and Public Health Vol 4, Issue 4 (2015) pp. 492-495

[9] Jon T. Ingjaldsson, Jon C. Laberg, and Julian F. Thayer, "Reduced Heart Rate Variability in Chronic Alcohol Abuse: Relationship with Negative Mood, Chronic Thought Suppression, and Compulsive Drinking", Society of Biological Psychiatry, (2002), pp. 1427-1436

[10] Phyllis K. Stein, et. al., "Heart Rate Variability and Measure of Autonomic Tone", American Heart Journal, vol. 127 no. 5 (Sept. 1993) pp. 1376-1381

[11] Katsuyuki Murata, Philip J. Landrigan, and Shunichi Araki, "Effects of age, heart rate, gender, tobacco and alcohol ingestion on R-R interval variability in human ECG", Journal of the Autonomic Nervous System, 37 (1992) pp.199-206

[12] Mika P. Tarvainen and Juha-Pekka Niskanen, "Kubios HRV Analysis version 2.0 beta USER'S GUIDE", Biosignal Analysis and Medical Imaging Group, Department of Physics, University of Kuopio, Finland

[13] L. Ljung,"System identification toolbox," The Matlab user's guide, 2012. [Online] Available: http://radio.feld.cvut.cz/matlab/pdf_doc/ident/ident.pdf

[14] Hong He, Xiaowen Yan and Wei Wei, "Meridian ECG Information Transmission System Modeling Using NARX Neural Network" ,IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), 2016. [Online] Available:http://ieeexplore.ieee.org/document/7550775/

[15] Chung Kit Wu, et. al. "A Precise Drunk Driving Detection Using Weighted Kernel based on Electrocardiogram", *Sensors*. [Online]. 16(5), pp659. Available: http://www.mdpi.com/1424-8220/16/5/659/htm

[16] Kotsianntis, Sotiris B., et. al. "Supervised Machine Learning: A review of classification techniques" 3-24, (2007)

[17] G.-B. Huang, "What are Extreme Learning Machines? Filling the Gap between Frank Rosenblatt's Dream and John von Neumann's Puzzle," Cognitive Computation, vol. 7, pp. 263-278, 2015.

[18] R. Savitha, S. Suresh, H.J. Kim, "A Meta-Cognitive Learning Algorithm for an Extreme Learning Machine Classifier" Cognitive Computation, vol. 6, pp. 253-263, 2014.

[19] Andrew Ng, "Support Vector Machines", 2011. [Online] Available: http://cs229.stanford.edu/notes/cs229-notes3.pdf Accessed: 10-Feb-2016

[20] Andrew Ng, "The Simplified SMO Algorithm", 2012. [Online] Available: http://cs229.stanford.edu/materials/smo.pdf Accessed: 10-Feb-2016

[21] Ian Poole, "Opamp Notch Filter Circuit" [Online] Available: http://www.radio-electronics.com/info/circuits/opamp_notch_filter/opamp_notch_filter.php

[22] "Band Stop Filter" [Online] Available: http://www.electronics-tutorials.ws/filter/band-stop-filter.html

[23] Mr. Hrishikesh Limaye1, Mrs. V.V. Deshmukh2, "ECG Noise Sources and Various Noise Removal Techniques: A Survey", International Journal of Application or Innovation in Engineering & Management, Volume 5, Issue 2, (2016)

[24] Dingfei Ge, "Cardiac arrhythmia classification using autoregressive modelling". [Online] Available: http://biomedical-engineering-online.biomedcentral.com/articles/10.1186/1475-925X-1-5

[25] Branislav Vuksanovic & Mustafa Alhamdi, "AR-based Method for ECG Classification and Patient Recognition," International Journal of Biometrics and Bioinformatics (IJBB), vol. 7 ,Issue 2, 2013

APPENDIX – A

Datasheets:

ANALOG DEVICES

Low Cost Low Power Instrumentation Amplifier

AD620

FEATURES

Easy to use
- Gain set with one external resistor (Gain range 1 to 10,000)
- Wide power supply range (±2.3 V to ±18 V)
- Higher performance than 3 op amp IA designs
- Available in 8-lead DIP and SOIC packaging
- Low power, 1.3 mA max supply current

Excellent dc performance (B grade)
- 50 μV max, input offset voltage
- 0.6 μV/°C max, input offset drift
- 1.0 nA max, input bias current
- 100 dB min common-mode rejection ratio (G = 10)

Low noise
- 9 nV/√Hz @ 1 kHz, input voltage noise
- 0.28 μV p-p noise (0.1 Hz to 10 Hz)

Excellent ac specifications
- 120 kHz bandwidth (G = 100)
- 15 μs settling time to 0.01%

APPLICATIONS

Weigh scales
ECG and medical instrumentation
Transducer interface
Data acquisition systems
Industrial process controls
Battery-powered and portable equipment

CONNECTION DIAGRAM

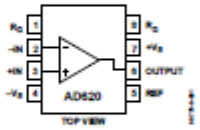


Figure 1. 8-Lead PDIP (N), CERDIP (Q), and SOIC (R) Packages

PRODUCT DESCRIPTION

The AD620 is a low cost, high accuracy instrumentation amplifier that requires only one external resistor to set gains of 1 to 10,000. Furthermore, the AD620 features 8-lead SOIC and DIP packaging that is smaller than discrete designs and offers lower power (only 1.3 mA max supply current), making it a good fit for battery-powered, portable (or remote) applications.

The AD620, with its high accuracy of 40 ppm maximum nonlinearity, low offset voltage of 50 μV max, and offset drift of 0.6 μV/°C max, is ideal for use in precision data acquisition systems, such as weigh scales and transducer interfaces. Furthermore, the low noise, low input bias current, and low power of the AD620 make it well suited for medical applications, such as ECG and noninvasive blood pressure monitors.

The low input bias current of 1.0 nA max is made possible with the use of Superβeta processing in the input stage. The AD620 works well as a preamplifier due to its low input voltage noise of 9 nV/√Hz at 1 kHz, 0.28 μV p-p in the 0.1 Hz to 10 Hz band, and 0.1 pA/√Hz input current noise. Also, the AD620 is well suited for multiplexed applications with its settling time of 15 μs to 0.01%, and its cost is low enough to enable designs with one in-amp per channel.

Table 1. Next Generation Upgrades for AD620

Part	Comment
AD8221	Better specs at lower price
AD8222	Dual channel or differential out
AD8226	Low power, wide input range
AD8220	JFET input
AD8228	Best gain accuracy
AD8295	+2 precision op amps or differential out
AD8429	Ultra low noise

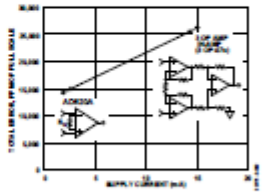


Figure 2. Three Op Amp IA Designs vs. AD620

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700 www.analog.com
Fax: 781.326.8703

SPECIFICATIONS

Typical @ 25°C, V_S = ±15 V, and R_L = 2 kΩ, unless otherwise noted.

Table 2.

		AD620A			AD620B			AD620S[1]			
Parameter	**Conditions**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Unit**
GAIN	G = 1 + (49.4 kΩ/R_G)										
Gain Range		1		10,000	1		10,000	1		10,000	
Gain Error[2]	V_{OUT} = ±10 V										
G = 1			0.03	0.10		0.01	0.02		0.03	0.10	%
G = 10			0.15	0.30		0.10	0.15		0.15	0.30	%
G = 100			0.15	0.30		0.10	0.15		0.15	0.30	%
G = 1000			0.40	0.70		0.35	0.50		0.40	0.70	%
Nonlinearity	V_{OUT} = −10 V to +10 V										
G = 1–1000	R_L = 10 kΩ		10	40		10	40		10	40	ppm
G = 1–100	R_L = 2 kΩ		10	95		10	95		10	95	ppm
Gain vs. Temperature											
	G = 1			10			10			10	ppm/°C
	Gain >1[2]			−50			−50			−50	ppm/°C
VOLTAGE OFFSET	(Total RTI Error = V_{OSI} + V_{OSO}/G)										
Input Offset, V_{OSI}	V_S = ±5 V to ± 15 V		30	125		15	50		30	125	μV
Overtemperature	V_S = ±5 V to ± 15 V			185			85			225	μV
Average TC	V_S = ±5 V to ± 15 V		0.3	1.0		0.1	0.6		0.3	1.0	μV/°C
Output Offset, V_{OSO}	V_S = ±15 V		400	1000		200	500		400	1000	μV
	V_S = ± 5 V			1500			750			1500	μV
Overtemperature	V_S = ±5 V to ± 15 V			2000			1000			2000	μV
Average TC	V_S = ±5 V to ± 15 V		5.0	15		2.5	7.0		5.0	15	μV/°C
Offset Referred to the Input vs. Supply (PSR)	V_S = ±2.3 V to ±18 V										
G = 1		80	100		80	100		80	100		dB
G = 10		95	120		100	120		95	120		dB
G = 100		110	140		120	140		110	140		dB
G = 1000		110	140		120	140		110	140		dB
INPUT CURRENT											
Input Bias Current			0.5	2.0		0.5	1.0		0.5	2	nA
Overtemperature				2.5			1.5			4	nA
Average TC			3.0			3.0			8.0		pA/°C
Input Offset Current			0.3	1.0		0.3	0.5		0.3	1.0	nA
Overtemperature				1.5			0.75			2.0	nA
Average TC			1.5			1.5			8.0		pA/°C
INPUT											
Input Impedance											
Differential			10||2			10||2			10||2		GΩ||pF
Common-Mode			10||2			10||2			10||2		GΩ||pF
Input Voltage Range[2]	V_S = ±2.3 V to ±5 V	−V_S + 1.9		+V_S − 1.2	−V_S + 1.9		+V_S − 1.2	−V_S + 1.9		+V_S − 1.2	V
Overtemperature		−V_S + 2.1		+V_S − 1.3	−V_S + 2.1		+V_S − 1.3	−V_S + 2.1		+V_S − 1.3	V
	V_S = ± 5 V to ±18 V	−V_S + 1.9		+V_S − 1.4	−V_S + 1.9		+V_S − 1.4	−V_S + 1.9		+V_S − 1.4	V
Overtemperature		−V_S + 2.1		+V_S − 1.4	−V_S + 2.1		+V_S 2.1	−V_S + 2.3		+V_S − 1.4	V

AD620

Parameter	Conditions	AD620A Min	AD620A Typ	AD620A Max	AD620B Min	AD620B Typ	AD620B Max	AD620S[1] Min	AD620S[1] Typ	AD620S[1] Max	Unit
Common-Mode Rejection											
Ratio DC to 60 Hz with 1 kΩ Source Imbalance	V_{CM} = 0 V to ± 10 V										
G = 1		73	90		80	90		73	90		dB
G = 10		93	110		100	110		93	110		dB
G = 100		110	130		120	130		110	130		dB
G = 1000		110	130		120	130		110	130		dB
OUTPUT											
Output Swing	R_L = 10 kΩ										
	V_S = ±2.3 V to ± 5 V	$-V_S$ + 1.1		$+V_S$ – 1.2	$-V_S$ + 1.1		$+V_S$ – 1.2	$-V_S$ + 1.1		$+V_S$ – 1.2	V
Overtemperature		$-V_S$ + 1.4		$+V_S$ – 1.3	$-V_S$ + 1.4		$+V_S$ – 1.3	$-V_S$ + 1.6		$+V_S$ – 1.3	V
	V_S = ±5 V to ± 18 V	$-V_S$ + 1.2		$+V_S$ – 1.4	$-V_S$ + 1.2		$+V_S$ – 1.4	$-V_S$ + 1.2		$+V_S$ – 1.4	V
Overtemperature		$-V_S$ + 1.6		$+V_S$ – 1.5	$-V_S$ + 1.6		$+V_S$ – 1.5	$-V_S$ + 2.3		$+V_S$ – 1.5	V
Short Circuit Current			±18			±18			±18		mA
DYNAMIC RESPONSE											
Small Signal –3 dB Bandwidth											
G = 1			1000			1000			1000		kHz
G = 10			800			800			800		kHz
G = 100			120			120			120		kHz
G = 1000			12			12			12		kHz
Slew Rate		0.75	1.2		0.75	1.2		0.75	1.2		V/µs
Settling Time to 0.01%	10 V Step										
G = 1–100			15			15			15		µs
G = 1000			150			150			150		µs
NOISE											
Voltage Noise, 1 kHz	Total RTI Noise = $\sqrt{(e_{ni}^2) + (e_{no}/G)^2}$										
Input, Voltage Noise, e_{ni}			9	13		9	13		9	13	nV/√Hz
Output, Voltage Noise, e_{no}			72	100		72	100		72	100	nV/√Hz
RTI, 0.1 Hz to 10 Hz											
G = 1			3.0			3.0	6.0		3.0	6.0	µV p-p
G = 10			0.55			0.55	0.8		0.55	0.8	µV p-p
G = 100–1000			0.28			0.28	0.4		0.28	0.4	µV p-p
Current Noise	f = 1 kHz		100			100			100		fA/√Hz
0.1 Hz to 10 Hz			10			10			10		pA p-p
REFERENCE INPUT											
R_{IN}			20			20			20		kΩ
I_{IN}	V_{IN+}, V_{REF} = 0		50	60		50	60		50	60	µA
Voltage Range		$-V_S$ + 1.6		$+V_S$ – 1.6	$-V_S$ + 1.6		$+V_S$ – 1.6	$-V_S$ + 1.6		$+V_S$ – 1.6	V
Gain to Output		1 ± 0.0001			1 ± 0.0001			1 ± 0.0001			
POWER SUPPLY											
Operating Range[4]		±2.3		±18	±2.3		±18	±2.3		±18	V
Quiescent Current	V_S = ±2.3 V to ±18 V		0.9	1.3		0.9	1.3		0.9	1.3	mA
Overtemperature			1.1	1.6		1.1	1.6		1.1	1.6	mA
TEMPERATURE RANGE											
For Specified Performance		–40 to +85			–40 to +85			–55 to +125			°C

[1] See Analog Devices military data sheet for 883B tested specifications.
[2] Does not include effects of external resistor R_G.
[3] One input grounded. G = 1.
[4] This is defined as the same supply range that is used to specify PSR.

ANALOG DEVICES

Single-Lead, Heart Rate Monitor Front End

Data Sheet **AD8232**

FEATURES

- Fully integrated single-lead ECG front end
- Low supply current: 170 µA (typical)
- Common-mode rejection ratio: 80 dB (dc to 60 Hz)
- Two or three electrode configurations
- High signal gain (G = 100) with dc blocking capabilities
- 2-pole adjustable high-pass filter
- Accepts up to ±300 mV of half cell potential
- Fast restore feature improves filter settling
- Uncommitted op amp
- 3-pole adjustable low-pass filter with adjustable gain
- Leads off detection: ac or dc options
- Integrated right leg drive (RLD) amplifier
- Single-supply operation: 2.0 V to 3.5 V
- Integrated reference buffer generates virtual ground
- Rail-to-rail output
- Internal RFI filter
- 8 kV HBM ESD rating
- Shutdown pin
- 20-lead 4 mm × 4 mm LFCSP package

APPLICATIONS

- Fitness and activity heart rate monitors
- Portable ECG
- Remote health monitors
- Gaming peripherals
- Biopotential signal acquisition

FUNCTIONAL BLOCK DIAGRAM

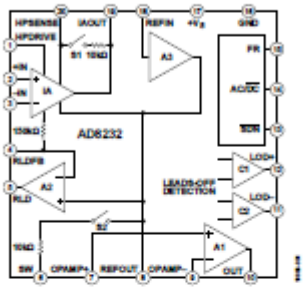


Figure 1.

GENERAL DESCRIPTION

The AD8232 is an integrated signal conditioning block for ECG and other biopotential measurement applications. It is designed to extract, amplify, and filter small biopotential signals in the presence of noisy conditions, such as those created by motion or remote electrode placement. This design allows for an ultralow power analog-to-digital converter (ADC) or an embedded microcontroller to acquire the output signal easily.

The AD8232 can implement a two-pole high-pass filter for eliminating motion artifacts and the electrode half-cell potential. This filter is tightly coupled with the instrumentation architecture of the amplifier to allow both large gain and high-pass filtering in a single stage, thereby saving space and cost.

An uncommitted operational amplifier enables the AD8232 to create a three-pole low-pass filter to remove additional noise. The user can select the frequency cutoff of all filters to suit different types of applications.

To improve common-mode rejection of the line frequencies in the system and other undesired interferences, the AD8232 includes an amplifier for driven lead applications, such as right leg drive (RLD).

The AD8232 includes a fast restore function that reduces the duration of otherwise long settling tails of the high-pass filters. After an abrupt signal change that rails the amplifier (such as a leads off condition), the AD8232 automatically adjusts to a higher filter cutoff. This feature allows the AD8232 to recover quickly, and therefore, to take valid measurements soon after connecting the electrodes to the subject.

The AD8232 is available in a 4 mm × 4 mm, 20-lead LFCSP package. Performance is specified from 0°C to 70°C and is operational from −40°C to +85°C.

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700
Technical Support www.analog.com

SPECIFICATIONS

V_S = 3 V, V_{REF} = 1.5 V, V_{CM} = 1.5 V, T_A = 25°C, FR=low, SDN=high, AC/$\overline{DC}$ = low, unless otherwise noted.

Table 1.

Parameter	Symbol	Test Conditions/Comments	Min	Typ	Max	Unit
INSTRUMENTATION AMPLIFIER						
Common-Mode Rejection Ratio, DC to 60 Hz	CMRR	V_{CM} = 0.35 V to 2.85 V, V_{DIFF} = 0 V	80	86		dB
		V_{CM} = 0.35 V to 2.85 V, V_{DIFF} = ±0.3 V		80		dB
Power Supply Rejection Ratio	PSRR	V_S = 2.0 V to 3.5 V	76	90		dB
Offset Voltage (RTI)	V_{OS}					
Instrumentation Amplifier Inputs				3	8	mV
DC Blocking Input[1]				5	50	µV
Average Offset Drift						
Instrumentation Amplifier Inputs				10		µV/°C
DC Blocking Input[1]				0.05		µV/°C
Input Bias Current	I_B			50	200	pA
		T_A = 0°C to 70°C		1		nA
Input Offset Current	I_{OS}			25	100	pA
		T_A = 0°C to 70°C		1		nA
Input Impedance						
Differential				10‖7.5		GΩ‖pF
Common Mode				5‖15		GΩ‖pF
Input Voltage Noise (RTI)						
Spectral Noise Density		f = 1 kHz		100		nV/√Hz
Peak-to-Peak Voltage Noise		f = 0.1 Hz to 10 Hz		12		µV p-p
		f = 0.5 Hz to 40 Hz		14		µV p-p
Input Voltage Range		T_A = 0°C to 70°C	0.2		$+V_S$	V
DC Differential Input Range	V_{DIFF}		−300		+300	mV
Output						
Output Swing		R_L = 50 kΩ	0.1		$+V_S$ − 0.1	V
Short-Circuit Current	I_{SET}			6.3		mA
Gain	A_V			100		V/V
Gain Error		V_{OUT} = 0 V		0.4		%
		V_{OUT} = −300 mV to +300 mV		1	3.5	%
Average Gain Drift		T_A = 0°C to 70°C		12		ppm/°C
Bandwidth	BW			2		kHz
RFI Filter Cutoff (Each Input)				1		MHz
OPERATIONAL AMPLIFIER (A1)						
Offset Voltage	V_{OS}			1	5	mV
Average TC		T_A = 0°C to 70°C		5		µV/°C
Input Bias Current	I_B			100		pA
		T_A = 0°C to 70°C		1		nA
Input Offset Current	I_{OS}			100		pA
		T_A = 0°C to 70°C		1		nA
Input Voltage Range			0.1		$+V_S$ − 0.1	V
Common-Mode Rejection Ratio	CMRR	V_{CM} = 0.5 V to 2.5 V		100		dB
Power Supply Rejection Ratio	PSRR			100		dB
Large Signal Voltage Gain	A_{VO}			110		dB
Output Voltage Range		R_L = 50 kΩ	0.1		$+V_S$ − 0.1	V
Short-Circuit Current Limit	I_{SET}			12		mA
Gain Bandwidth Product	GBP			100		kHz
Slew Rate	SR			0.02		V/µs
Voltage Noise Density (RTI)	e_n	f = 1 kHz		60		nV/√Hz
Peak-to-Peak Voltage Noise (RTI)	$e_{n\,p-p}$	f = 0.1 Hz to 10 Hz		6		µV p-p
		f = 0.5 Hz to 40 Hz		8		µV p-p

MICROCHIP

MCP3004/3008

2.7V 4-Channel/8-Channel 10-Bit A/D Converters with SPI Serial Interface

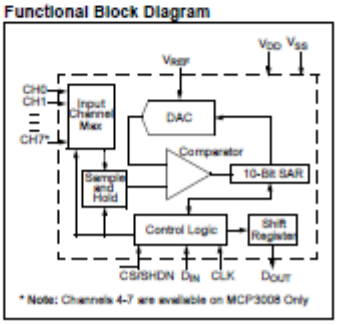
Features

- 10-bit resolution
- ± 1 LSB max DNL
- ± 1 LSB max INL
- 4 (MCP3004) or 8 (MCP3008) input channels
- Analog inputs programmable as single-ended or pseudo-differential pairs
- On-chip sample and hold
- SPI serial interface (modes 0,0 and 1,1)
- Single supply operation: 2.7V - 5.5V
- 200 ksps max. sampling rate at V_{DD} = 5V
- 75 ksps max. sampling rate at V_{DD} = 2.7V
- Low power CMOS technology
- 5 nA typical standby current, 2 µA max.
- 500 µA max. active current at 5V
- Industrial temp range: -40°C to +85°C
- Available in PDIP, SOIC and TSSOP packages

Applications

- Sensor Interface
- Process Control
- Data Acquisition
- Battery Operated Systems

Functional Block Diagram



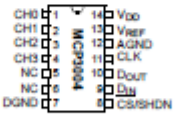
* Note: Channels 4-7 are available on MCP3008 Only

Description

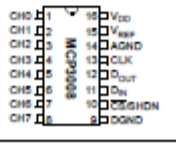
The Microchip Technology Inc. MCP3004/3008 devices are successive approximation 10-bit Analog-to-Digital (A/D) converters with on-board sample and hold circuitry. The MCP3004 is programmable to provide two pseudo-differential input pairs or four single-ended inputs. The MCP3008 is programmable to provide four pseudo-differential input pairs or eight single-ended inputs. Differential Nonlinearity (DNL) and Integral Nonlinearity (INL) are specified at ±1 LSB. Communication with the devices is accomplished using a simple serial interface compatible with the SPI protocol. The devices are capable of conversion rates of up to 200 ksps. The MCP3004/3008 devices operate over a broad voltage range (2.7V - 5.5V). Low-current design permits operation with typical standby currents of only 5 nA and typical active currents of 320 µA. The MCP3004 is offered in 14-pin PDIP, 150 mil SOIC and TSSOP packages, while the MCP3008 is offered in 16-pin PDIP and SOIC packages.

Package Types

PDIP, SOIC, TSSOP



PDIP, SOIC



1.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings †

V_{DD} 7.0V
All Inputs and Outputs w.r.t. V_{SS} ..... −0.6V to V_{DD} + 0.6V
Storage Temperature ..... −65°C to +150°C
Ambient temperature with power applied ..... −65°C to +150°C
Soldering temperature of leads (10 seconds) ..... +300°C
ESD Protection On All Pins (HBM) ..... ≥ 4 kV

† **Notice:** Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operational listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL SPECIFICATIONS

Electrical Characteristics: Unless otherwise noted, all parameters apply at V_{DD} = 5V, V_{REF} = 5V, T_A = -40°C to +85°C, f_{SAMPLE} = 200 ksps and f_{CLK} = 18*f_{SAMPLE}. Unless otherwise noted, typical values apply for V_{DD} = 5V, T_A = +25°C.

Parameter	Sym	Min	Typ	Max	Units	Conditions
Conversion Rate						
Conversion Time	t_{CONV}	—	—	10	clock cycles	
Analog Input Sample Time	t_{SAMPLE}		1.5		clock cycles	
Throughput Rate	f_{SAMPLE}	—	—	200 75	ksps ksps	V_{DD} = V_{REF} = 5V V_{DD} = V_{REF} = 2.7V
DC Accuracy						
Resolution			10		bits	
Integral Nonlinearity	INL	—	±0.5	±1	LSB	
Differential Nonlinearity	DNL	—	±0.25	±1	LSB	No missing codes over temperature
Offset Error		—	—	±1.5	LSB	
Gain Error		—	—	±1.0	LSB	
Dynamic Performance						
Total Harmonic Distortion		—	-76		dB	V_{IN} = 0.1V to 4.9V@1 kHz
Signal-to-Noise and Distortion (SINAD)		—	61		dB	V_{IN} = 0.1V to 4.9V@1 kHz
Spurious Free Dynamic Range		—	78		dB	V_{IN} = 0.1V to 4.9V@1 kHz
Reference Input						
Voltage Range		0.25	—	V_{DD}	V	Note 2
Current Drain		—	100 0.001	150 3	µA µA	 $\overline{CS}$ = V_{DD} = 5V
Analog Inputs						
Input Voltage Range for CH0 or CH1 in Single-Ended Mode		V_{SS}	—	V_{REF}	V	
Input Voltage Range for IN+ in pseudo-differential mode		IN-	—	V_{REF}+IN-		
Input Voltage Range for IN- in pseudo-differential mode		V_{SS}-100	—	V_{SS}+100	mV	

Note 1: This parameter is established by characterization and not 100% tested.
2: See graphs that relate linearity performance to V_{REF} levels.
3: Because the sample cap will eventually lose charge, effective clock rates below 10 kHz can affect linearity performance, especially at elevated temperatures. See **Section 6.2 "Maintaining Minimum Clock Speed"**, "Maintaining Minimum Clock Speed", for more information.

MCP3004/3008

ELECTRICAL SPECIFICATIONS (CONTINUED)

Electrical Characteristics: Unless otherwise noted, all parameters apply at V_{DD} = 5V, V_{REF} = 5V, T_A = -40°C to +85°C, f_{SAMPLE} = 200 ksps and f_{CLK} = 18*f_{SAMPLE}. Unless otherwise noted, typical values apply for V_{DD} = 5V, T_A = +25°C.

Parameter	Sym	Min	Typ	Max	Units	Conditions
Leakage Current		—	0.001	±1	µA	
Switch Resistance		—	1000	—	Ω	See Figure 4-1
Sample Capacitor		—	20	—	pF	See Figure 4-1
Digital Input/Output						
Data Coding Format			Straight Binary			
High Level Input Voltage	V_{IH}	0.7 V_{DD}	—	—	V	
Low Level Input Voltage	V_{IL}		—	0.3 V_{DD}	V	
High Level Output Voltage	V_{OH}	4.1	—	—	V	I_{OH} = -1 mA, V_{DD} = 4.5V
Low Level Output Voltage	V_{OL}	—	—	0.4	V	I_{OL} = 1 mA, V_{DD} = 4.5V
Input Leakage Current	I_{LI}	-10	—	10	µA	V_{IN} = V_{SS} or V_{DD}
Output Leakage Current	I_{LO}	-10	—	10	µA	V_{OUT} = V_{SS} or V_{DD}
Pin Capacitance (All Inputs/Outputs)	C_{IN}, C_{OUT}	—	—	10	pF	V_{DD} = 5.0V (**Note 1**) T_A = 25°C, f = 1 MHz
Timing Parameters						
Clock Frequency	f_{CLK}	—	—	3.6 1.35	MHz MHz	V_{DD} = 5V (**Note 3**) V_{DD} = 2.7V (**Note 3**)
Clock High Time	t_{HI}	125	—	—	ns	
Clock Low Time	t_{LO}	125	—	—	ns	
$\overline{CS}$ Fall To First Rising CLK Edge	t_{SUCS}	100	—	—	ns	
$\overline{CS}$ Fall To Falling CLK Edge	t_{CSD}	—	—	0	ns	
Data Input Setup Time	t_{SU}	50	—	—	ns	
Data Input Hold Time	t_{HD}	50	—	—	ns	
CLK Fall To Output Data Valid	t_{DO}	—	—	125 200	ns ns	V_{DD} = 5V, See Figure 1-2 V_{DD} = 2.7V, See Figure 1-2
CLK Fall To Output Enable	t_{EN}	—	—	125 200	ns ns	V_{DD} = 5V, See Figure 1-2 V_{DD} = 2.7V, See Figure 1-2
$\overline{CS}$ Rise To Output Disable	t_{DIS}	—	—	100	ns	See Test Circuits, Figure 1-2
$\overline{CS}$ Disable Time	t_{CSH}	270	—	—	ns	
D_{OUT} Rise Time	t_R	—	—	100	ns	See Test Circuits, Figure 1-2 (**Note 1**)
D_{OUT} Fall Time	t_F	—	—	100	ns	See Test Circuits, Figure 1-2 (**Note 1**)

Note 1: This parameter is established by characterization and not 100% tested.
2: See graphs that relate linearity performance to V_{REF} levels.
3: Because the sample cap will eventually lose charge, effective clock rates below 10 kHz can affect linearity performance, especially at elevated temperatures. See **Section 6.2 "Maintaining Minimum Clock Speed"**, "Maintaining Minimum Clock Speed", for more information.

AD8232 Data Sheet

Parameter	Symbol	Test Conditions/Comments	Min	Typ	Max	Unit
RIGHT LEG DRIVE AMPLIFIER (A2)						
Output Swing		R_L = 50 kΩ	0.1		$+V_S$ − 0.1	V
Short-Circuit Current	I_{SHT}			11		mA
Integrator Input Resistor			120	150	180	kΩ
Gain Bandwidth Product	GBP			100		kHz
REFERENCE BUFFER (A3)						
Offset Error	V_{OS}	R_L > 50 kΩ		1		mV
Input Bias Current	I_B			100		pA
Short-Circuit Current Limit	I_{SHT}			12		mA
Voltage Range		R_L = 50 kΩ	0.1		$+V_S$ − 0.7	V
DC LEADS OFF COMPARATORS						
Threshold Voltage				$+V_S$ − 0.5		V
Hysteresis				60		mV
Propagation Delay				0.5		µs
AC LEADS OFF DETECTOR						
Square Wave Frequency	F_{AC}		50	100	175	kHz
Square Wave Amplitude	I_{AC}			200		nA p-p
Impedance Threshold		Between +IN and −IN	10	20		MΩ
Detection Delay				110		µs
FAST RESTORE CIRCUIT						
Switches		S1 and S2				
On Resistance	R_{ON}		8	10	12	kΩ
Off Leakage				100		pA
Window Comparator						
Threshold Voltage		From either rail		50		mV
Propagation Delay				2		µs
Switch Timing Characteristics						
Feedback Recovery Switch On Time	t_{SW1}			110		ms
Filter Recovery Switch On Time	t_{SW2}			55		ms
Fast Restore Reset	t_{RST}			2		µs
LOGIC INTERFACE						
Input Characteristics						
Input Voltage (AC/$\overline{DC}$ and FR)						
Low	V_{IL}			1.24		V
High	V_{IH}			1.35		V
Input Voltage ($\overline{SDN}$)						
Low	V_{IL}			2.1		V
High	V_{IH}			0.5		V
Output Characteristics		LOD+ and LOD− terminals				
Output Voltage						
Low	V_{OL}			0.05		V
High	V_{OH}			2.95		V
SYSTEM SPECIFICATIONS						
Quiescent Supply Current				170	230	µA
		T_A = 0°C to 70°C		210		µA
Shutdown Current				40	500	nA
		T_A = 0°C to 70°C		100		nA
Supply Range			2.0		3.5	V
Specified Temperature Range			0		70	°C
Operational Temperature Range			−40		+85	°C

[1] Offset referred to the input of the instrumentation amplifier inputs. See the Input Referred Offsets section for additional information.

APPENDIX – B

Akarsh N Kolekar (1PI13EC009)

Hardware:

1. Selection of low noise, high gain, Instrumentation Amplifier IC
2. Study of Circuit – 1 (Two stage Op-Amp Circuit)
3. Re-soldering of Notch Circuit Design – 3

Software:

1. Implementation of ELM Classifier (MATLAB and Python)
2. Study of Meta Cognitive ELM
3. Python GUI for Real Time System

APPENDIX – C

Apoorv Vatsal (1PI13EC017)

Hardware:

1. Set-Up of Raspberry Pi
2. Interface of ADC MCP 3008
3. Interface of AD8232 with Raspberry Pi
4. Implemented signal capture code to save as CSV file on Raspberry Pi

Software:

1. Implementation of Wavelet Decomposition for pre-processing
2. Implementation of Frequency Feature Extraction (MATLAB and Python)
3. Implementation of Autoregressive Modelling with Exogenous Inputs
4. Python GUI for Real Time System

APPENDIX – D

Rakshith Vishwanatha (1PI13EC075)

Hardware:

1. Study of LPF, HPF filter ranges and Q-Factor for band reject filters
2. Design of Circuit – 3 (Notch Filter Circuit)
3. Soldering of Notch Circuit Design – 3 and MCP 3008
4. Solved 1kHz sampling issue on Raspberry Pi

Software:

1. Implementation of IIR filtering for pre-processing
2. Implementation of Time Domain Feature Extraction (MATLAB and Python)
3. Implementation of Non-Linear Feature Extraction (MATLAB)
4. Implementation of SVM classifier