Imperial College London
Department of Electrical and Electronic Engineering

Computationally-Efficient Algorithms for Real-Time ECG Baseline Removal

Onur Guven

15 June 2016

Supervised by Dr. Timothy Constandinou

Submitted for the degree of
Doctor of Philosophy and the Diploma of Imperial College London

Declaration

I hereby declare that this thesis and the work described is my own and that everything else is appropriately referenced and attributed.

Onur Guven

Copyright Declaration

Acknowledgement

No matter how much I appreciate and be thankful to all who have had an influence on my work, I do not think I can truly express how grateful I am. I know for a fact that their insight and contributions to this work and to my development cannot be limited to this page only.

Firstly and foremost, I must thank my supervisor Dr. Timothy Constandinou for his contributions, ideas, patience, support and welcoming attitude towards me. He has been a great mentor and a role model. I cannot express how thankful I am as his envision and organised approach have helped me through struggles and doubts throughout my journey, and always acted as a friend more than a supervisor.

I would like to thank Dr. Amir Eftekhar for his support, ideas, discussions and friendship. Even though his name is not listed as a supervisor, I believe he has contributed to my work more than a co-supervisor and I am truly grateful for his presence throughout my journey. As one of the main contributors, I would also like to thank Dr. Wilko Kindt for his ideas, contributions and his valuable time. Although the direction of my PhD. has changed throughout, he has always been there for me when I needed help and guidance.

My family has always been my greatest support throughout this whole process and at times when I was struggling to keep up my motivation, they have been there to give me the much-needed last "push". I cannot express how grateful I am as they have been with me throughout life's many obstacles never doubting, always supportive.

Best friends have always been there for me throughout my journey. Their support and encouragement are truly appreciated. I have to thank Doga Ozturk in particular, as he has taken his valuable time from his PhD. and helped me in proof-reading.

I would like to thank Texas Instruments Corporation for their partial financial support and also Dr. Reza Hoshyar and Dr. Giovanni Frattini for their ideas.

My dear colleagues in Neural Interfaces Group and in Centre for Bio-Inspired Technology can not be forgotten. I would like to thank them all for their friendship and insightful discussions and in particular name Dr. Deren Barsakcioglu, Dr. Lieuwe Leene, Dr. Nicoletta Nicolaou, Dr. Song Luan, Dr. Mohammedreza Sohbati, Dr. Yan Liu, Dr. Benjamin Evans and Timo Lauteslager for their contributions.

Last but not least, I would like to thank Izabela Wojcicka-Grezsiak and Wiesia Hsissen for their efforts in helping everyone in our group to their best.

Abstract

ECG signal processing has been one of the major studied topics in the biomedical field. The introduction of new techniques and the extensions to the previous keep constantly evolving the span of the ECG research, providing a true realisation of the problems specific to each new approach. For this reason, thorough data analysis and accuracy evaluation have been the most significant tools in effective quantifying of ECG noise elimination techniques.

The ECG signal is generally defined to have a spectral content between 50 mHz and 150 Hz with a few millivolts in amplitude, and identified as susceptible to physiological and environmental interferences. The elimination of noise interferences, in particular the baseline wander, is a major concern in preserving the ECG signal integrity (i.e the ST segment) due to the overlapping spectral content of noise sources with this segment. The inherent complexity of such a problem has led to computationally-intensive algorithms in the literature (i.e. Empirical Mode Decomposition (EMD), Independent Component Analysis (ICA), Wavelet Transforms (WT) and others) and the removal of the baseline drift is acquired off-line with powerful simulation tools. The adaptations of these methods for ambulatory designs, on the other hand, demonstrate substantial accuracy degradation due to scaling. Therefore, real-time approaches to match comparable accuracy to the computational intensive algorithms are yet to be proposed.

This research investigates a computationally-efficient baseline wander removal technique and targets comparable performance to its computational off-line counterparts reported in the literature while preserving the signal integrity of the ECG.

This work introduces a novel hardware-efficient real-time baseline estimation method based on three distinctive "isoelectric" fiducial point detections per heart beat. These detected points are cubic spline interpolated to achieve a realistic representation of the baseline estimation, and removed from the noisy signal to provide an "undistorted" ECG signal representation. Computational efficiency of this approach is further improved with a novel weighted piecewise linear interpolation technique. This approach targets non-uniformly sampled systems with less computational requirements compared to the higher order polynomial interpolation. The MCU-based real-time hardware system realisation of these algorithms demonstrates accurate ambulatory system response and this is the first tested system level design addressing baseline wander removal with detailed analysis. The validated tests have presented original contributions for baseline wander detection and removal by tackling one of the most crucial challenges currently present in clinically valid ECG signal processing. The accuracy and the computational requirements of the developed algorithms show real-time capabilities of the overall system and challenge its computational ECG signal processing counterparts.

Dedicated to my beloved parents
for their endless love and support...

Contents

List of Figures

List of Tables

List of Abbreviations

AAMI - Association for the Advancement of Medical Instrumentation

ADC - Analogue to Digital Converter

AFE - Analogue Front End

AHA - American Heart Association

AM - Active Mode

AMI - Acute Myocardial Infarction

ANSI - American National Standards Institute

AV - Atrioventricular Node

CCS - Code Composer Studio

CMRR - Common Mode Rejection Ratio

CVD - Cardiovascular Disease

DSP - Digital Signal Processor

DWT - Discrete Wavelet Transform

ECG - Electrocardiogram

EMD - Empirical Mode Decomposition

EMG - Electromyogram

EMI - Electromagnetic Interference

ESC - European Society of Cardiology

FFT - Fast Fourier Transform

FIR - Finite Impulse Response

FPGA - Field Programmable Gate Array

FRAM - Ferroelectric Random Access Memory

GCC - GNU Compiler Collection

HRV - Heart Rate Variability

HPF - High Pass Filter

IA - Instrumentation Amplifier

ICA - Independent Component Analysis

IDE - Integrated Development Environment

IEC - International Electrochemical Commission

IIR - Infinite Impulse Response

IMF - Intrinsic Mode Function

ISR - Interrupt Service Routine

LMS - Least Mean Squares

LPF - Low Pass Filter

LPM - Low Power Mode

MAF - Moving Average Filter

MCU - Microcontroller Unit

MIT-BIH - Massachusettes Institute of Technology and Boston's Beth Isral Hospital

NSTEMI - Non-ST Segment Elevated Myocardial Infarction

PCHIP - Piecewise Cubic Hermite Interpolation

QVR - Quadratic Variation Reduction

RLS - Recursive Least Squares

RMSD - Root Mean Square Deviation

RRS - Recursive Running Sums

RSA - Respiration Sinus Arrhythmia

SA - Sinoatrial Node

SNR - Signal to Noise Ratio

SOS - Second-order Sections

STEMI - ST Segment Elevated Myocardial Infarction

UART - Universal Asynchronous Receiver/Transmitter

WAF - Wavelet Adaptive Filter

WFDB - Waveform Database

WPL - Weighted Piecewise Linear

Chapter 1

Introduction

Electrophysiology, a field first studied in the second half of the 18th century [1], investigates the electrical properties of biological cells and tissues. With the advent of technology, it has been shown how these bio-potential reactions convey crucial information about human nature more so than previously predicted.

These electrical interactions of tissues and cells cover the whole human body and control voluntary and involuntary responses to the surroundings such as the electrical activity of the heart, the contractions and relaxations of certain muscle groups or the activity of nerve impulses within the brain. Fig. 1.1 compares the signal characteristics of such common bio-potential signals [2]. It can be observed that several of these signals have characteristics that overlap in both amplitude (level) and frequency content (bandwidth). In general, bio-potential signals span a frequency range from 50 mHz up to 10 kHz and signal amplitudes span from microvolt to millivolt levels. Interference can thus be attributed to overlapping signal bands with other bio-potential signals (a kind of "bio-crosstalk") but also to external sources. Therefore, several approaches have been reported to eliminate the different sources of interference in order to obtain an "undistorted" signal that is viable for clinical diagnosis.

Based on the characteristics of physiological signals, certain requirements need to be addressed. Due to their low frequency content, large time constants are necessary to filter out noise interferences while preserving the signal integrity. In order to achieve those large time constants, analogue solutions require large resistances and capacitances. Usually, this is not a design problem with discrete components; however, integrated solutions with large capacitance values are difficult to be fabricated due to large die area requirements. On the other hand, large DC offsets and powerline interferences from the mains disturb the signal quality as the amplitude of the physiological signals is in microvolt to millivolt ranges. These noise sources require efficient filtering techniques and low noise analogue front end (AFE) designs to avoid crucial information loss and achieve high dynamic ranges.

With such small amplitude levels and bio-crosstalk interferences, the electrical signals transmitted through the corresponding type of sensors or transducers need to be filtered and amplified by the front-end designs before further processing takes place in the digital domain. One of the challenges in the front end designs is the requirement of the low power read out circuits for long-

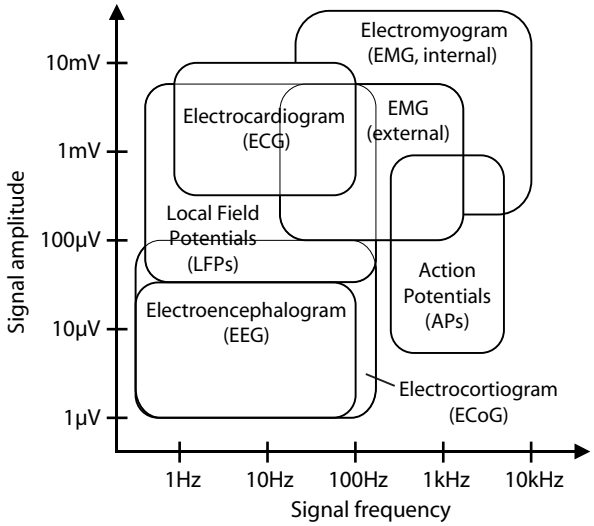


Figure 1.1.: Biopotential signals and their characteristics

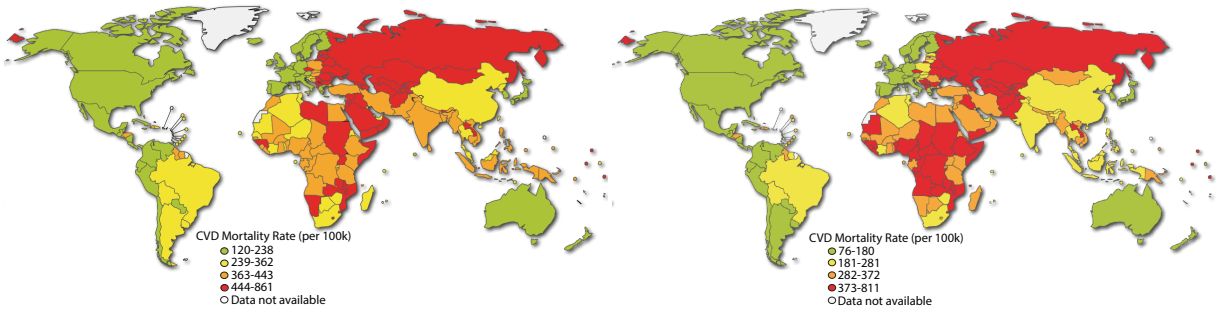
term ambulatory operations. Unless paid attention, long term operations via body patches can cause necrosis (death of living cells) in the muscle tissue due to the excessive heat dissipation of integrated designs. Seese *et al.* state that a heat flux of $0.08\,\mathrm{W/cm^2}$ is enough to cause the death of biological cells adjacent to the heat source [3, 4]. In addition to that, increased power dissipation would require change of batteries limiting its continuous time ambulatory operation. Therefore, power dissipation should always be considered in system level design.

1.1. Motivation

Every day thousands of people face life-threatening situations based on cardiovascular diseases. These instances not only result in negative emotional impact both for the patient and their family, but also have socio-economic consequences such as requiring life-long treatment and/or medicine intake to reduce a future heart failure.

According to the World Health Organisation's year 2012 statistics, an estimated 17.5 million people (31 % of the global deaths) died from cardiovascular diseases (CVD) and CVDs are still the main cause of deaths globally [5]. These reported deaths are mainly observed in low and middle income countries (82 % of all the CVDs), equally distributed between men and women. Fig. 1.2 displays the cardiovascular death rates of male and female patients around the globe. The same organisation predicts by the year 2030, 23.6 million people will die from CVDs mainly because of heart diseases and stroke annually [6]. In addition, the prediction also states that CVDs are projected to remain the single leading cause of death by the year 2030 [7]. Therefore, the need to improve modern healthcare systems for the reliable diagnosis and early detection of CVDs is certainly a priority.

With the advent of medical device technology, mobile and ambulatory applications prove to be the new advancement in pre-detection of coronary heart diseases and many others. Therefore, there is an increasing demand by both professionals and patients in shifting from hospitalised



(a) Male (b) Female

Figure 1.2.: World Health Organization's CVD related death rates. Extracted from [8]

care solutions to home care detection systems in order to act before heart disorders reach critical levels. On the other hand, with the development of home care solutions fewer sensor measurements cause less electrode irritation when compared to conventional 12-lead hospitalized solutions. Once critical levels are detected via ambulatory devices, further tests can be held in hospitals by keeping electrode irritation to a minimum level. If such systems are deployed at home or integrated into lifestyle, there are further challenges such as compliance, good electrode placement and general reliability.

Fig. 1.3 shows the examples of the easily available devices in the ECG market. These systems can be divided into two main categories as consumer and medical-grade electronics. In the recent years, there has been a growing trend towards the field of wearable consumer electronics devices

Figure 1.3.: ECG devices available on the market. Extracted from [9–12]

for lifestyle and sports monitoring; however, these "lifestyle" devices do not generate clinically valid data. On the other hand, in medical-grade electronics Smartheart states to be the "first" and "only" 12-lead ECG device on the market that enables the detection of heart attacks by requiring telemetry and diagnosis at a telemedicine center [13]. With the limited progress in wireless technology, the total amount of data transfer in ambulatory devices is restricted. High resolution data transmission combined with high sampling rates dissipate too much energy, given the capacity of a typical battery. Secondly, the progress in battery technology has been very limited. Therefore, seeking for new methodologies to alert the patient about their condition is critical.

1.2. Challenge on ECG systems - Defining the problem

Various ECG systems are available on the market for commercial use and still loads of research have been conducted on ECG read out circuits and processing methods. Fig. 1.4 shows the target application of these systems which deploy different algorithms and hardware design to process raw ECG data with varying success rates. According to a study by 8 cardiologists, nine different algorithms yielded correct classifications ranging from 69.7 % to 76.3 % success rate [15]. Although this experiment was performed in 1991 and improved algorithms with the "self learning" attributes have been utilised in the following years, diagnosis of acute cardiac ischaemia have been a challenge throughout the years. In 2001, another study showed a sensitiv-

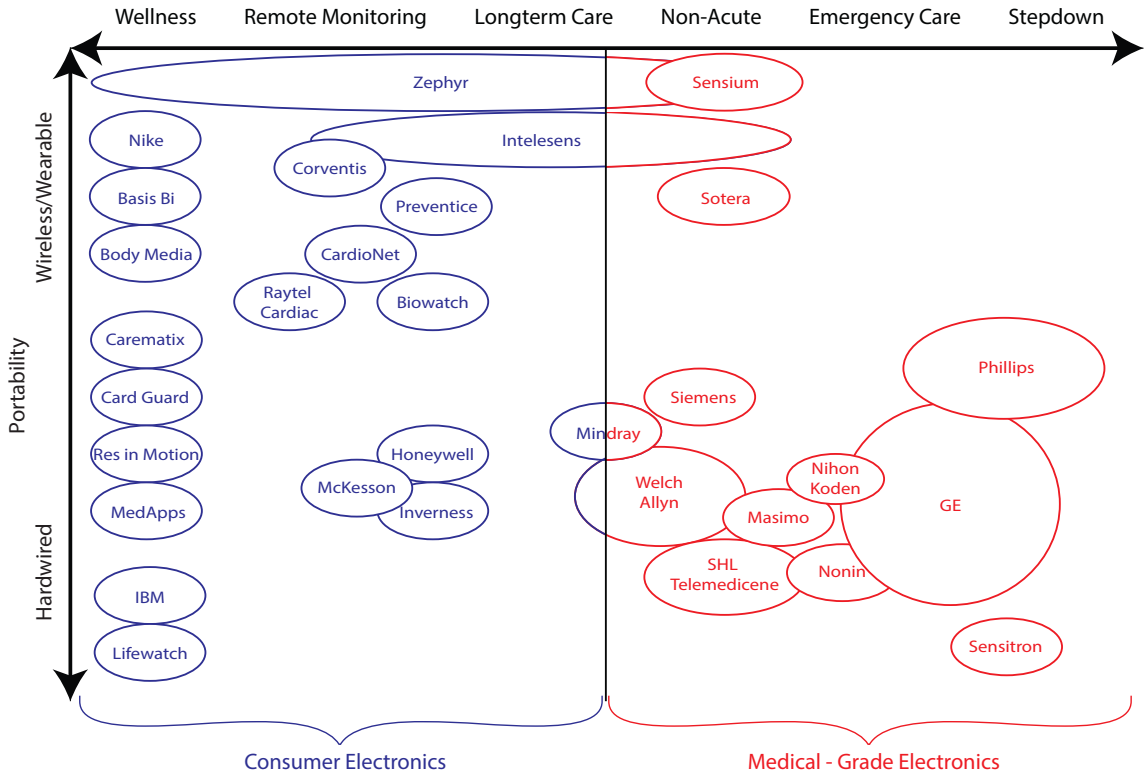


Figure 1.4.: ECG healthcare monitoring space. Redrawn from [14]

ity of 76 % and a specificity of 88 % by a computer algorithm in the identification of ventricular late potentials [16]. These results demonstrate that identification of myocardial infarction is challenging due to the high noise levels present in the signal of interest. Therefore, American Heart Association and International Electrotechnical Commission required high resolution readings for diagnosis purposes within the ECG frequency band in the evaluation of ST segment morphology thoroughly [17–20].

The evaluation of the baseline drift in addition to other noise sources and their removal carries a significant importance to all ECG measurements. However, since the frequency of interest of the ST segment coincides with the frequency content of the baseline wander, removing the baseline wander through a high-pass filter often results in elevation or in suppression of the ST segment [20, 21]. However, in order to evaluate the readings correctly, baseline wander has to be removed in a way without affecting the ST segment. Various approaches with different success rates do exist, nevertheless there is no consensus on the best methodology due to ECG recordings varying from person to person and even from one instance to another. Most of the methods reported in the literature are computational and applied for hospital care solutions only due to high power dissipation requirements.

Most of the ambulatory ECG systems like commercial Holter monitors feature low resolution readings from 8-bits to 12-bits in their specifications. In addition, stored data in those systems need to be processed by software algorithms eventually, resulting in hospitalised care solutions rather than home care. On the other hand, real-time systems for HR-ECGs mainly digitise the signal of interests with higher resolutions and/or utilise advanced signal processing techniques to overcome the lack of SNR. However, as ventricular potentials are in microvolt levels, computerised solutions like signal averaging of a few hundreds of heart beats can still produce errors due to the large noise transitions and/or noisy beats [22]. Adopting these solutions to ambulatory care is not easy due to the low resolution systems not identifying certain problems and high resolution systems requiring high process power dependant computerised solutions. These computerised solutions limit the overall systems to be applicable to hospital care purposes only due to the high power requirements of the system.

1.3. Research Hypothesis

As indicated in the previous section, ECG baseline wander removal poses significant challenges in clinically valid data interpretation. Elimination of this noise source through conventional filtering and/or signal processing techniques, results in ST segment elevation/depression due to its frequency content coinciding with the noise interference. This disturbs the ECG signal integrity. Therefore, ambulatory diagnostics require frequency-independent and computationally-efficient ECG baseline wander removal.

1.4. Research Objectives

Various works have been reported in the literature related with this area and different approaches exist in removing the noise interferences mainly caused by respiration, muscle movement, improper electrode site preparation and deficient electrodes. Most of these systems are designed for hospital care solutions and with advancement of home care and ambulatory applications, predetection of coronary heart diseases gains significant importance. Therefore, specific research objectives to test the Hypothesis are:

- To detect and remove baseline wander interference as required by the applicable standards while accomplishing resource efficient real-time system design. To achieve this goal, noise source characteristics affecting the in-band signal quality have to be comprehended thoroughly as they will be forming the main challenges of system level design. These noise interferences have to be isolated and/or removed in baseline drift estimation and the detection algorithm has to be agile to predict and perform real-time measurements without requiring significantly large number of cycles and data storage as these would be limiting real-time system implementation.

- To investigate methods for reducing complexity in the most computationally demanding parts of baseline detection algorithm (i.e. interpolation). Comprehensive analysis shows that even though polynomial approaches achieve smoother fits, sometimes they are not accurate and they rely on computationally demanding number of operations, thus limiting their real-time implications. This project develops a novel and an efficient algorithm to utilise the information acquired from baseline detection stage improving the accuracy vs complexity trade off in ECG applications.

- To implement baseline wander detection and interpolation algorithms in an embedded system to achieve real-time baseline wander estimation. This way, computational complexity of the overall system such as dedicated memory, and hardware requirements are investigated, and the power consumption of the overall system is quantified.

1.5. Outline of the Thesis

The remainder of the Thesis is organised as follows:

- Chapter 2 focuses on ECG characteristics, heart activity and noise artefacts present in ECG signal as well as the challenges associated with removing these artefacts. The discussion then follows on with a detailed literature review covering conventional systems, and the state-of-the-art techniques in baseline wander removal.

- Chapter 3 discusses a novel hardware efficient approach for ECG baseline drift removal that is shown to preserve integrity of the ST segment by tracking 3 “isoelectric” points within the ECG waveform. The hypotheses behind such an approach and the proposed

methodology are covered in detail along with the in depth analysis with synthetic and real test results. These findings are then compared and evaluated with the state-of-the art techniques.

- Chapter 4 introduces a computationally-efficient interpolation method that has been optimised for use in ECG baseline drift removal algorithm. A feasibility study investigates the trade-offs between computational complexity and accuracy of the proposed two-stage interpolation approach and compares its evaluated synthetic and real test results with higher order polynomial interpolation techniques.
- Chapter 5 tests both algorithms in an embedded target presenting and evaluating the measured results. To follow up, complexity measure of both algorithms is quantified in more detail and an in depth analysis in regards to power consumption is presented.
- Chapter 6 concludes the thesis highlighting original contributions in addition to possible future directions.

Chapter 2

Background & The State-of-the-Art

The previous chapter has introduced the key physiological signals and their susceptibility to noise interferences briefly. As discussed, these electrical activities are small in amplitude and their frequency content often overlaps with environmental noise sources and other physiological signals. The removal of these noise sources poses a significant challenge such that without reliable signal processing techniques, the processed output often deforms in the process. This deformation results in clinically significant information to be irrecoverable before the data is even interpreted by a clinician.

Baseline wander has been a vastly studied subject in ECG signal recording especially in hospitalised care solutions [23,24]. These clinical systems focus on preserving the signal integrity while addressing the noise activity through extensively computational digital signal processing techniques. On the other hand, real-time applications in ambulatory care do not trifle ECG signal integrity. Usually, these systems implement system-on-chip solutions and address baseline drift with filtering techniques. As a result, the signal of interest gets distorted and its clinical validity is undermined within the process. Removal of these noise sources while preserving the signal quality in real-time applications, therefore, remains unsolved, and whether or not computationally efficient algorithmic techniques ascertain these have to be investigated in more detail.

This chapter focuses on the ECG morphology and the state-of-the-art techniques in baseline wander removal. Section 2.1 provides a brief background on ECG signals and their properties. These cover basic ECG morphology knowledge to detect and estimate the baseline wander without distorting the signal of interest. Later, these are going to form the basis of the baseline wander estimation technique developed in the following chapters. In Section 2.2, the state-of-the-art review details up-to-date methods in ECG baseline drift removal and provides a brief background about each technique. The issues and problems that are currently associated with each method in baseline wander detection and their real-time implementation suitability are also discussed.

2.1. ECG Background

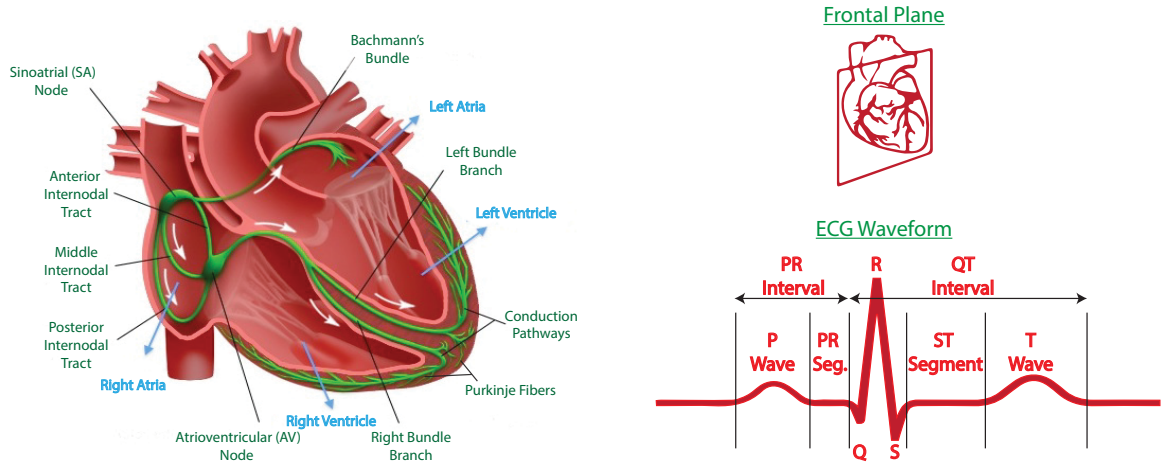
Electrocardiography (ECG) is the transthoracic interpretation of the electrical impulse of the heart. It is generated by the right atrium of the heart at a site called the sinoatrial node. Electrocardiograms record this activity by means of electrodes attached to the outer surface of the chest over a period of time in a non-invasive procedure. In the sections that follow, a background to heart activity, lead placements, ECG characteristics and noise interference types are explained.

2.1.1. Heart Activity

The heart is composed of two major types of cardiac muscle cells, the cardiomyocytes and the cardiac pacemaker cells [25]. These two types of cells differ from each other and fulfil different purposes during the electrical activity of the heart: (1) The former are responsible for the mechanical movement of the heart and form the bulk of the cells present in the atria and ventricles ($\approx 99\,\%$ [26]); whereas (2) the latter generate and conduct the electrical impulses through the heart and are significantly fewer in number.

Each heart beat starts with the depolarisation of cardiac pacemaker cells at the sinoatrial (SA) node, which is located in the posterior and anterior walls of the right atrium. These autorhythmic cells generate the electrical impulses and are responsible for sinus rhythm as the induction spreads through the heart. This propagation is illustrated in Fig. 2.1(a) with white arrows.

Prior to the atrial systole (contraction), the blood flows through the atrium to the ventricles passively. The first wave, namely the P wave, that is shown in Fig. 2.1(b) is formed when the sinoatrial node discharges and the depolarisation impulse spreads over the atria through Bach-



(a) Electrical system of the heart. Adapted from [27] (b) Frontal plane and typical ECG waveform

Figure 2.1.: Conducting components of the heart pathway including the sinoatrial node, the internodal pathways, the atrioventricular node, the right and left bundle branches, and the Purkinje fibers with frontal plane and typical ECG waveform

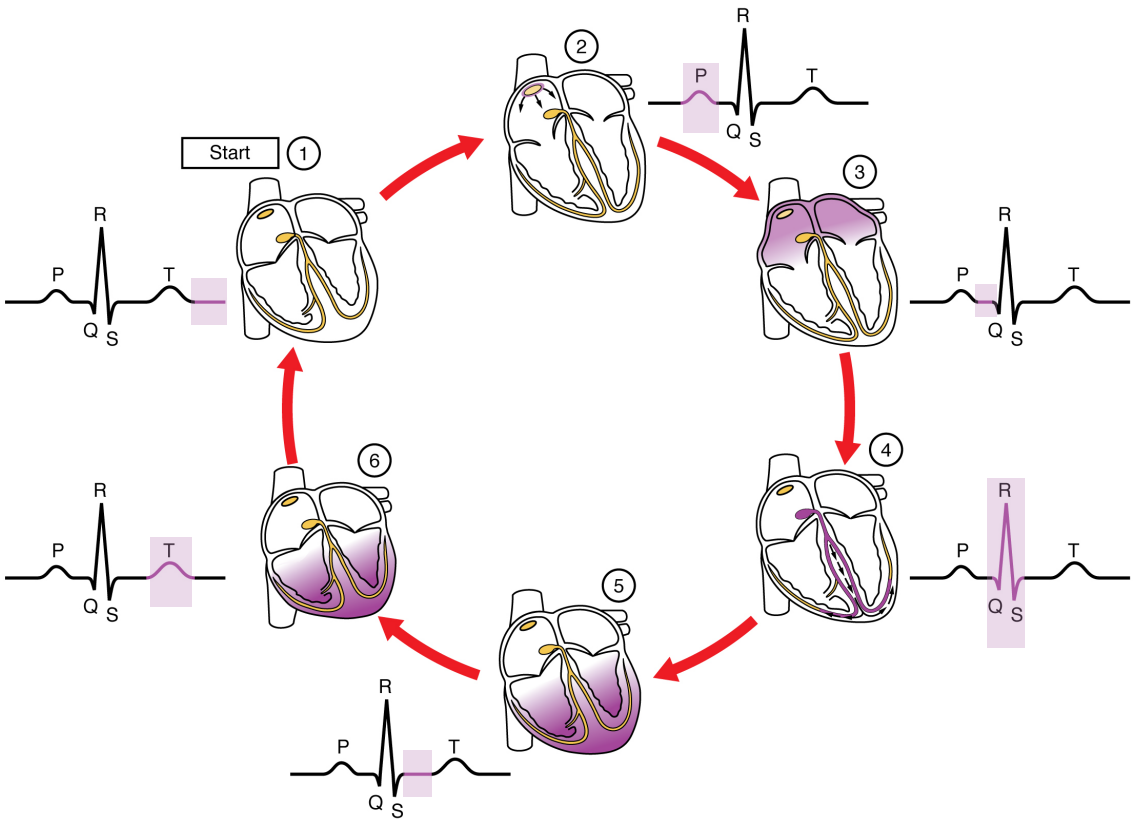


Figure 2.2.: ECG waveform tracings with the electrical and mechanical events of a heart contraction. Extracted from [26]

mann's Bundle and internodal pathways (anterior, middle and posterior tracts). An illustration of this SA node activity is shown in Fig. 2.2(1), and (2).

As the impulse reaches to the second clump of autorhythmic cells at the atrioventricular (AV) node located in the inferior section of the right atrium, there is a delay in the completion of pumping blood in the atria as shown in Fig. 2.2(3). This brief pause, referred to as PR segment, tops off the ventricles with blood, which increases the ventricular pressure. As a result of this pressure change, the AV valves close to prevent blood flowing backwards and the semilunar valves (aortic and pulmonary valves) open.

The impulse is then transmitted to the atriaventricular bundle through conduction pathways and branches into Purkinje fibers as illustrated in Fig. 2.2(4). During this time interval, the QRS complex is generated and the impulse is spread over the ventricles causing the large muscle mass to depolarise. This depolarisation continues on until the pressure within the ventricles equalises with the pressure in the aorta and pulmonary arteries.

After the contraction of the ventricles, the heart goes into a silent phase called the ST segment. During this time interval, no electrical activity can be passed through the myocardium as shown in Fig. 2.2(5). Any electrical activity in that interval can be interpreted as a cause of myocardial infarction or ischaemic behaviour.

Finally, the T-wave is generated when the ventricles repolarise; in other words, when the ventricular myocardium relaxes as depicted in Fig. 2.2(6). During the diastole, the pressure in the aorta and pulmonary arteries exceeds the pressure in the ventricles resulting in the semilunar valves to close. After the completion of ventricle repolarisation, the heart prepares for upcoming contractions.

As can be seen from the Fig 2.1(b), there does not seem to be a recharging phase for the atria. Since the total muscle mass in ventricles is heavier and the relaxation of atria occurs when the ventricles depolarise, the missing waveform is concealed beneath the QRS complex. Additionally, some papers report another wave after the T wave called the U wave [28]. However, the origin of the U wave is uncertain and its possible causes are thought to be interventricular septal repolarisation or slow ventricular repolarisation.

2.1.2. Table of Cardiac Events

Based on the heart activity described previously, cardiac events with typical ECG feature characteristics are presented in Table 2.1. Additionally, the illustration of each specific event is depicted in Fig. 2.1(b) and this table lists possible problems associated with each segment that can lead into sinus arrest, atrial enlargement, myocardial infarction, ischaemia and many others. The reader can refer to the work of Andrew R Houghton *et al.* for further descriptive analysis on ECG disorders associated with each certain cardiac event markers [28]. Throughout the thesis, ECG events and their typical durations listed in this table are used in estimating the ECG baseline wander.

Table 2.1.: Typical cardiac event durations [28]

ECG Event	**Cardiac Event**	**Problems**	**Typical Values**
P wave	Atrial Depolarisation	Absent, inverted, tall or wide	$< 0.25\,\text{mV}$ in amp. $< 0.12\,\text{s}$ in dur.
PR interval	Time from atrial to ventricular depolarisation	Longer or shorter duration than typical values	$0.12\,\text{s} < \text{x} < 0.2\,\text{s}$ in dur. Should be consistent
QRS complex	Ventricular depolarisation	Abnormal shape or tall, small or wide complex	$< 2.5\,\text{mV}$ in amp. $< 0.12\,\text{s}$ in dur.
ST segment	Pause in electrical activity	Elevated or depressed	ST level shifts $< 0.1\,\text{mV}$
T wave	Ventricular repolarisation	Tall small or inverted	Half the size of QRS
QT interval	Time between ventricular de & re-polarisation	Longer or shorter duration than typical values	$\text{QTc} > 0.44\,\text{s}$ $\text{QTc} < 0.35\,\text{s}$

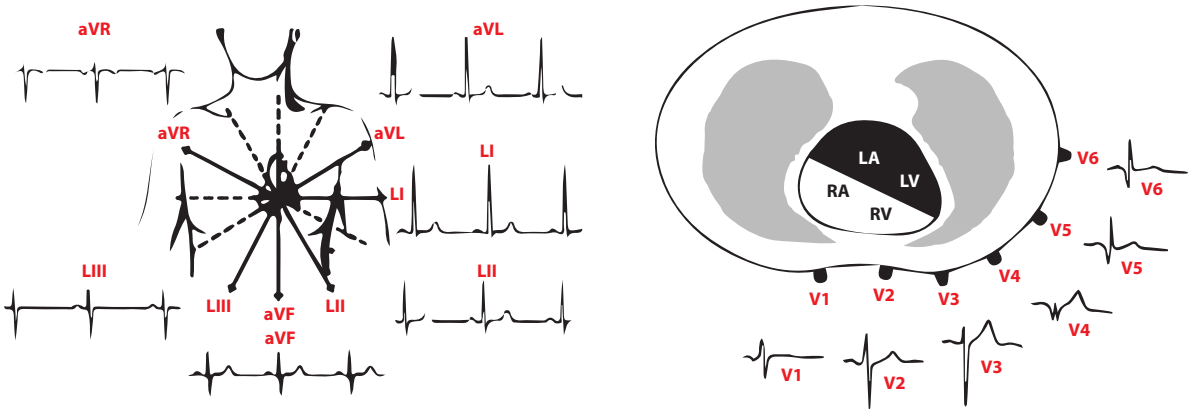
2.1.3. Lead Nomenclature

The ECG waveform is composed of 3 main waves namely the P-/T- waves and the QRS complex. When these waves are investigated from different leads, a wide range of abnormalities regarding the electrical conduction system and the muscle tissue of the heart's pumping chambers can be diagnosed based on the typical values tabulated in Table 2.1. Therefore, it is important to understand the "lead" arrangements to make sense of the ECG waveforms properly.

Leads capture electrical activity of the heart from different viewpoints. Therefore, more lead placements provide a more comprehensive picture of the heart's electrical activity from different angles. A conventional 12 lead ECG collects information via six limb and six chest leads as shown in Fig 2.3(a) and 2.3(b) respectively. Limb leads in Fig 2.3(a) are abbreviated as LI, LII, LIII, aVR, aVL and aVF, whereas six chest leads in Fig 2.3(b) are abbreviated from V1 to V6.

Limb leads (LI, LII and LIII) are utilised to form the Einthoven's triangle by placing the electrodes at the ankles. This orientation forms an inverted equilateral triangle with the heart located at the centre generating zero potential when the voltages are summed. An example of such lead placement at the corners of the chest is illustrated in Fig 2.4, as well as the corresponding limb lead equations demonstrating their arrangements as in Eq. 2.1, 2.2, and 2.3. In these equations, left arm electrode is abbreviated as LA, right arm electrode is abbreviated as RA and left leg electrode is abbreviated as LL.

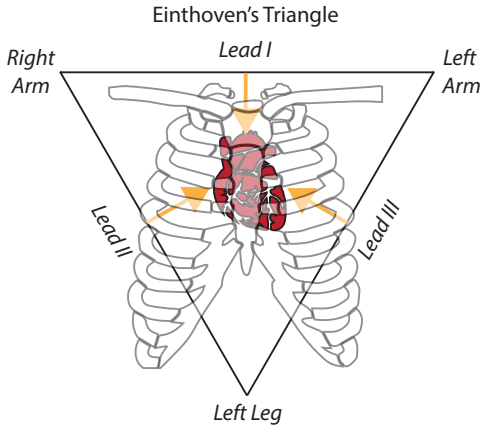
ECG leads are categorised as bipolar (LI, LII, and LIII) or unipolar (augmented and chest leads). Bipolar leads measure the voltage difference between two distinctive points, whereas unipolar leads utilise a reference point. This reference point, V_W, is often generated by a Wilson central terminal through a simple resistive network to obtain a potential average across the body as noted in Eq. 2.4, where left arm, right arm and left leg electrodes are denoted as LA, RA and LL respectively. Utilising this reference point, V_W, with the right arm, left arm and left leg electrodes, augmented limb leads, aVR, aVL and aVF are derived as in Eq. 2.5, 2.6, and 2.7 respectively. These leads see the heart from different angles compared to the limb leads



(a) 6 Limb leads (3 normal + 3 augmented)

(b) 6 Chest leads

Figure 2.3.: 12-Lead ECG placement. Redrawn from [29]



$$\text{LI} = LA - RA \tag{2.1}$$

$$\text{LII} = LL - RA \tag{2.2}$$

$$\text{LIII} = LL - LA \tag{2.3}$$

Equation Abbrevations:
LI = Lead I, LII = Lead II,
LIII = Lead III, LA = Left Arm,
RA = Right Arm, LL = Left Leg

Figure 2.4.: Placement of limb leads and limb lead equations. Redrawn from [30].

due to the vectorial formation from one electrode to the Wilson central terminal.

$$\text{V}_W = \frac{1}{3} * (RA + LA + LL) \tag{2.4}$$

$$\text{aVR} = \frac{3}{2} * (RA - V_W) \tag{2.5}$$

$$\text{aVL} = \frac{3}{2} * (LA - V_W) \tag{2.6}$$

$$\text{aVF} = \frac{3}{2} * (LL - V_W) \tag{2.7}$$

Every single lead captures a different signal depending on the motion of the electrical current relative to the lead positioning. This way a certain type of deflection is observed on the ECG trace as illustrated in Fig. 2.5. An example of a positive deflection can be observed in Lead-II recordings as a result of electrical impulses moving towards the lead with ventricular depolarisation. This way, all six limb leads form the "hexaxial reference system" and the heart's electrical activity in the frontal plane is observed from different angles [31].

As the lead types define the typical ECG waveforms, a detection algorithm is required to assess lead orientations and define reference points in system level design. In the event of a myocardial infarction, the ECG trace displays changes in the leads looking at that region, namely Lead-II,-III and aVF [28]. Based on this, the baseline detection algorithm is going to focus on Lead-II recordings as Lead-III trace generates marginally smaller amplitudes. Lead-II recordings display positive deflections in all P-/T- waves and QRS complexes and do not require Wilson central terminal during implementation.

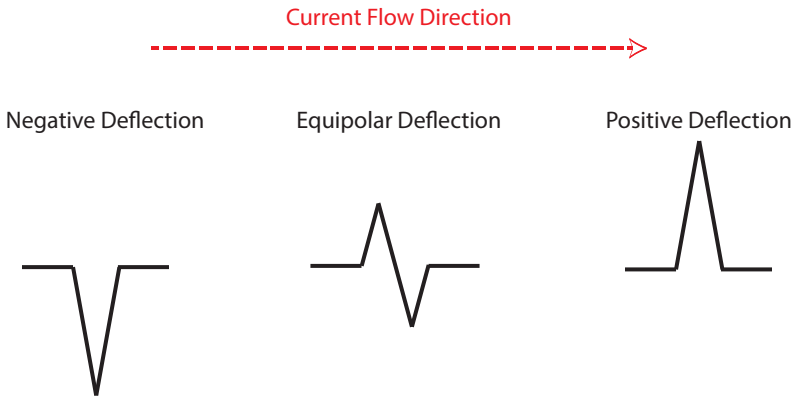


Figure 2.5.: Deflection type based on current flow direction

2.1.4. Myocardial Infarction

As covered in Section 1.2, the removal of noise interferences poses a significant challenge in preserving ST segment integrity. Evaluation of this time interval yields clinically valid information in acute myocardial infarction (AMI) diagnosis. This section briefly describes the symptoms of AMI and the subsequent changes observed in a Lead-II ECG trace.

Myocardial infarction is the name of the condition when the blood flow halts to a part of the heart. This condition occurs when coronary arteries are occluded due to an unstable build up of white blood cells, cholesterol and fat causing oxygen deprivation in the heart tissue [31]. Due to these prolonged ischaemic conditions, necrosis of myocardial cells occur within a period of time [32]. Therefore, timely diagnosis of myocardial infarctions is key to preventing life threatening arrhythmias [33].

There are two categories of AMI which differentiate from each other based on the elevation and depression observed at the ST segment of the ECG trace [31]. In ST elevated myocardial infarction (STEMI) cases, the ST segment elevation occurs due to major damage of heart muscles. In a non-ST segment elevated myocardial infarction (NSTEMI), patients have a partial blockage of the major coronary artery or a full blockage of minor ones resulting in an ST depression.

Fig. 2.6 illustrates both of the STEMI and NSTEMI responses. In STEMI, there are typically three changes evolving in time over a period of minutes to hours. These changes initiate with ST segment elevation followed by T wave inversion and Q wave formation [31]. On the other hand, patients experiencing NSTEMI with acute coronary syndrome display ST depression and T wave inversion [32, 34]. Both cases require myocardial infarction treatments according to the clinical practice guidelines [35].

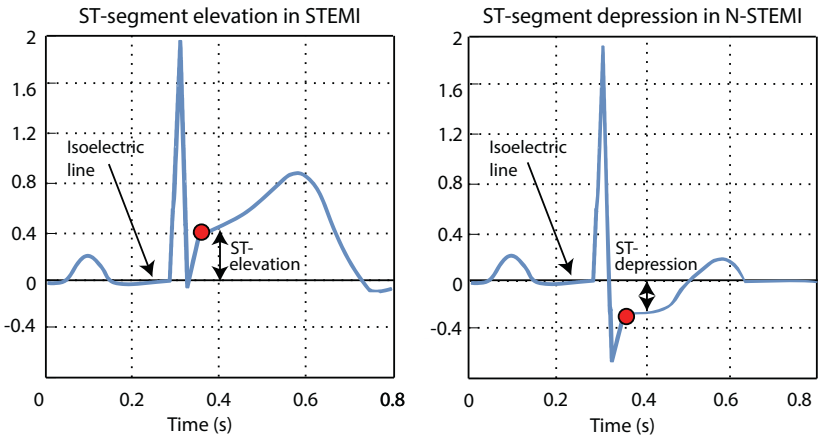


Figure 2.6.: Types of ST elevation and depression seen in myocardial infarction

2.1.5. ECG Characteristics & Noise Artefacts

ECG Characteristics ECG signals are in the order of a few millivolts as illustrated in Fig. 1.1. These signals are patient-specific and the amplitude differences depend on factors such as the total muscle mass of the chambers and their de-/re- polarisation states. Each QRS complex can reach up to 2.5 millivolts, and they are significantly larger in amplitude compared to P and T waves. Typical values of each of these waves are presented in Table 2.1.

Heart rate varies with emotional and physiological conditions, and heart rates from 60 beats/min to 100 beats/min are considered to be normal [36]. As the heart rate gets faster or slower, excessive conditions such as tachycardia or bradycardia occur respectively. In the literature, bradycardia is defined for heart rates below 60 beats per minute [37]; however, elderly people, athletes or even normal people during deep sleep can experience lower heart rates in rare cases. For that reason, heart rates of 40 beats/min and below are considered as absolute bradycardia [28].

The lowest frequency component of an ECG waveform is defined based on the heart rates. American Heart Association (AHA) states heart rates below 30 bpm (0.5 Hz) are "unlikely" whereas heart rates below 40 bpm (0.67 Hz) are "uncommon" in practice and recommends non-linear phase response filters to have a maximum cut-off frequency of 0.05 Hz to avoid ST segment distortion [20, 21]. This requirement is relaxed to 0.67 Hz and below for filters with no phase distortion whereas high frequency content up to 150 Hz is required for diagnostic purposes [38].

Noise Artefacts As interferences superimpose on to the signal of interest, the retrieval of crucial information by preserving the signal integrity becomes much of a challenge for small amplitude signals. These interferences and their characteristics are provided in this section.

ECG signals are weak bio-potential signals with low signal to noise ratio. Large DC offset, baseline wander, powerline interference, motion artefacts, defibrillation pulses, pace maker pulses and the electrical activity of skeletal muscles (EMG) interfere with the signal of interest. Elimination of these interferences poses several challenges due to improper electrode placements

Table 2.2.: Noise artefact types seen in Lead-II ECG recordings [40]

Noise artefact type	Maximum Amplitude	Frequency Range
Baseline drift	15 % of peak-to-peak (p-p) ECG	0.15 - 0.3 Hz depends on the respiration rate
Motion artefacts	500 % of p-p ECG	1 - 10 Hz lasts for 100 ms to 500 ms
Muscle contraction (EMG)	10 % of p-p ECG	DC - 10 kHz
Powerline interference	Up to 50 % of p-p ECG	50 / 60 Hz depends on the locale
Electrode contact noise	Max recorder output	50 / 60 Hz
Electrosurgical noise	200 % of p-p ECG	100 kHz - 1 MHz
Thermal noise	$\frac{kT}{C}$	Frequency dependent
Quantization noise	$SQNR \approx 1.76 + 6.02 * Q\, dB$	$\approx kf_s \pm f_{input}$

and filter applications [39]. Typical examples and their characteristics observed in a single channel lead ECG system (Lead-II) are listed in Table 2.2. Origins of each of these noise sources are detailed as follows:

- *Baseline wander* forms as a consequence of respiration of the patient. During inhalation, the chest expands and this movement results in an impedance change seen by the amplifier. Similarly, exhalation creates an effect in the opposite direction and this complete cycle generates baseline wander which can be modelled as a sinusoid. The fundamental frequency of this sinusoid is related to the respiration rate whereas its amplitude content varies with the lead positioning. In a Lead-II recording, the baseline and the amplitude variation can be approximated as 15% of the peak-to-peak (p-p) ECG signal [40].

- *Motion artefacts* are also caused by the electrode-skin impedance changes seen by the amplifier. These transient artefacts are attributed to the movements and the vibrations of the subject which often result in large fluctuations at the output. They can be modelled as biphasic signals lasting for 500 ms with a maximum amplitude level of 5 times the peak to peak ECG signal [40].

- *Muscle contractions* cause artefactual potentials due to neural excitation of muscle groups in the vicinity of the recording sites. Most of the power of EMG contractions is within 20 to 200 Hz range with their mean power point located below 100 Hz [41]. These transient bursts can be modelled as zero mean band limited Gaussian noise with their frequency content ranging from DC up to 10 kHz lasting for 50 ms [40]. The amplitude levels of these noise sources depend on the muscle mass and the fat surrounding the recording

area. Common analysis techniques of these amplitude levels involve running a root mean square of the signal over a short observational period of time [41].

- *Powerline interference* is the coupling of the frequency content of the mains to the patient's body due to parasitic capacitance between the patient and the power lines. Therefore, the frequency contents of these interferences depend on the geographical location and occur due to poor grounding and/or not appropriate filtering. This type of noise artefact requires high common mode rejection ratio (CMRR) at the front-end stages, active right leg drives and utilisation of notch filters.

- *Electrode contact noise* occurs due to loss of electrode-skin contact and is observed as step changes on the ECG trace. As the ECG signal is capacitively coupled, these disconnections occur as large artefacts with superimposed powerline interference. Electrode contact noise can be modelled as randomly occurring step changes, and decays exponentially to the baseline [40].

- *Electrosurgical noise* interferes with ECG signal during surgery. This radio frequency signal with extraordinary large transient voltages applied to the patient's skin surface requires adaptive filtering to acquire a "clean" ECG signal [42]. Even though the frequency bands are completely different, an aliased version of these type of noise interference corrupts the ECG signal. The amplitude, duration and the aliased frequency characteristics of such interferences vary and aliasing depends on the sampling frequency of the ECG system.

- *Thermal noise* occurs due to agitating thermal charge carriers generating a stochastic Gaussian noise distribution. This noise interference is mostly contributed by resistors or amplifiers in ECG systems and is proportional to the square root of the noise bandwidth as in Eq. 2.8, where k_B is the Boltzmann's constant in joules per kelvin, T is the resistor's absolute temperature in kelvin, R is the resistor value in ohms and Δf is the bandwidth in hertz over which the noise is measured. In an RC network, Eq. 2.8 simplifies to $\frac{kT}{C}$ when integrated over the bandwidth of the RC network as noted in Eq. 2.9.

$$\mathrm{v}_n = \sqrt{4k_B T R \Delta f} \tag{2.8}$$

$$\mathrm{v}_n^2 = 4k_B T R * \int_0^\infty H(\omega)\, d\omega = \frac{2k_B T}{\pi\, C} \tan^{-1}(\omega)\,|_0^\infty = \frac{kT}{C} \tag{2.9}$$

- *Quantisation noise* occurs during analogue to digital conversion as the real values are approximated with a finite set of discrete levels. These interferences have flat power spectral density and affect the system in a similar manner to an additive white noise [43]. The overall noise amplitude levels are determined by the number of quantisation bits, Q, and the general equation is noted in Table 2.2.

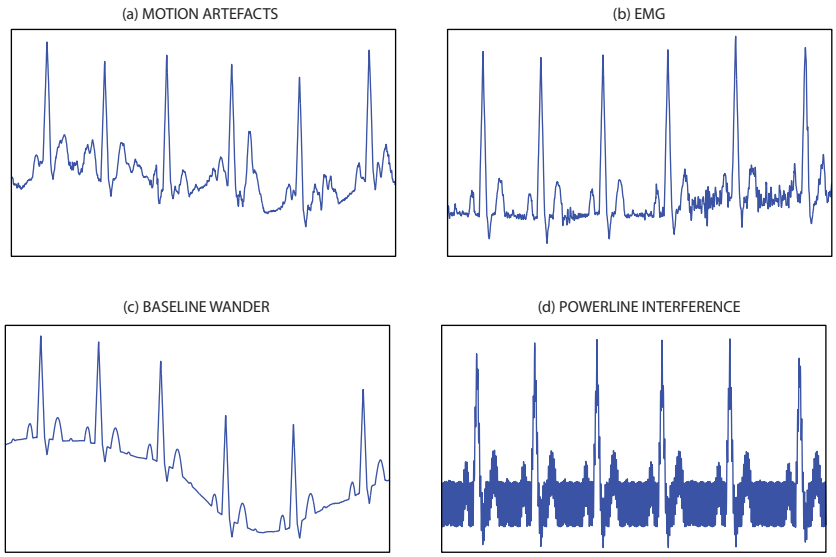


Figure 2.7.: Noise types superimposed on a synthetic ECG signal. Shown are: (a) Motion artefacts; (b) EMG interference; (c) Baseline wander; and (d) Powerline interference

Fig 2.7 displays noisy ECG instances showing motion artefacts, muscle contractions, baseline wander and powerline interference superimposed on a synthetic ECG.

2.2. The State-of-the-Art In ECG Baseline Removal

ECG systems are divided into two main categories based on their coupling methods, namely referred to as AC and DC coupled systems. Each system has its advantages and disadvantages associated with the noise artefact removal methods utilised thereafter. Here, common requirements for each system and their main difference are explained in detail.

Even though a sampling rate of twice the desired high frequency cut-off is required (Nyquist rate), 1990 AHA report states that sampling rates at 2 or 3 times the theoretical minimum are recommended [38]. Studies showed that a sampling rate of 500 Hz is needed to capture 150 Hz high frequency content to reduce the amplitude error measurements to 1 % in adults [44, 45]. This high frequency content is also mentioned by the American National Standards Institute and the Association for the Advancement of Medical Instrumentation (ANSI/AAMI) standard [46]. However, these sampling rates are not sufficient to capture pacemaker stimuli which are generally shorter than 0.5 ms in duration, therefore, oversampling is necessary to detect pacemaker pulses reliably [20].

Systemic noise sources affect both systems and standards define maximum allowances regarding each of these noise interferences. One such requirement is outlined for cable, circuit and display noise, which is mainly contributed by thermal noise. ANSI/AAMI allows a maximum of 30 μV_{p-p} at 150 Hz bandwidth during a 10 second of ECG recording [46]. As the error limitation is defined over a window and the white noise is often expressed in V_{RMS}, a conversion factor is calculated by means of the inverse cumulative distribution function of the yield from a standard

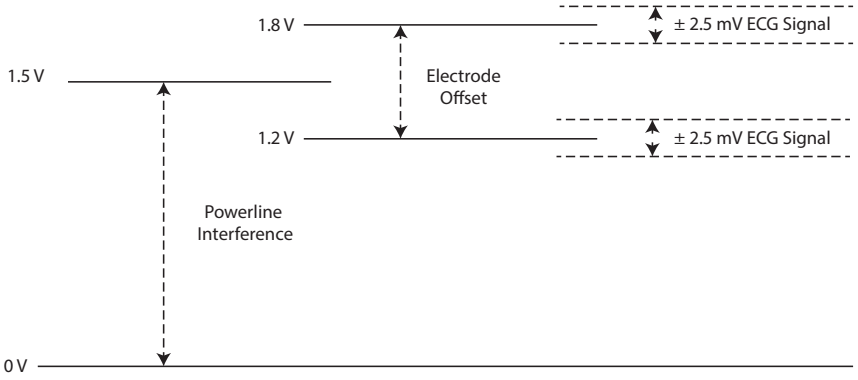


Figure 2.8.: Powerline interference and electrode offset in ECG systems

normal distribution in MATLAB. These parameters are defined as in the work of Forrest W. Breyfogle [47] and based on these definitions, yield of 1 in 5000 samples (at 500 Hz sampling over 10 second window) corresponds to a confidence level of $5\,\sigma$. As root mean square (RMS) is by definition equal to $1\,\sigma$, a maximum of $6\,\mu V_{RMS}$ is allowed at the specified sampling rate. Texas Instruments define the same requirement as $3.75\,\mu V_{RMS}$ at this sampling rate [48].

Similar to the thermal noise, ANSI/AAMI defines the requirements for common mode rejection and limits a maximum input referred output signal of $1\,mV_{p-p}$ signal over a 60 second window when $20\,V_{RMS}$ input signal is applied [46]. At a sampling rate of 500 Hz, this requirement corresponds to a confidence level of $5.5\,\sigma$ (1 part in 30000) and equates to 110 dB common mode rejection ratio.

The main difference in both techniques is based on the removal of the electrode offset and the low frequency content up to 0.05 Hz. Electrode offset forms across the electrode-skin interface due to the uneven distribution of anions and cations [2]. Characteristics of this half-cell potential are determined by the manufacturing material and standards require ECG waveforms to be displayed in the presence of $\pm 300\,mV$ electrode offset when applied to any lead [46]. Fig. 2.8 shows the dynamic range and the powerline interference present to both systems.

2.2.1. AC Coupled Systems

In AC coupled systems, removal of the electrode offset increases the effective dynamic range and relaxes the analogue-to-digital converter (ADC) resolution requirements. However, the large time constant requirement of the analogue high-pass filter at the front end is a challenge and the non-linear phase response of these filters must not distort the ST segment. Therefore, the high-pass cut-off frequency is defined up to 0.05 Hz as discussed in Section 2.1.5.

Conventional AC coupled ECG systems consist of an instrumentation amplifier (IA), a high-pass filter (HPF), an additional gain stage, an anti-aliasing low-pass filter (LPF) and a low resolution ADC implementation. Fig. 2.9 illustrates a typical implementation and the signal path of a single lead AC coupled ECG analogue front end (AFE). These systems require more analogue signal processing components compared to DC coupled systems and require low noise amplifiers to be utilised for system level design as the noise free dynamic range at the ADC must

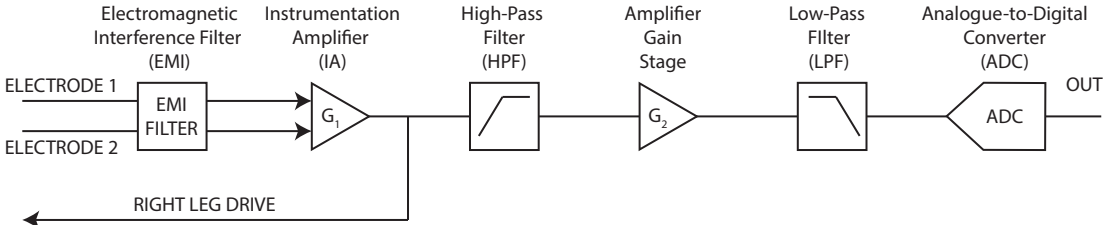


Figure 2.9.: A typical AC coupled front end architecture in ECG systems

comply with the maximum allowable noise provided by the standards as covered previously.

In these type of implementations, high CMRR of the instrumentation amplifier suppresses the common mode interference and the first gain stage amplifies the output with a small factor. Following this, a high-pass filter with its cut-off frequency defined below 0.05 Hz suppresses the electrode offset and the second gain stage utilises the full dynamic range of the ADC by amplifying the signal. The output of the second stage is then filtered by an anti aliasing low-pass filter with a cut-off frequency at 150 Hz to discard the out-of-band images to fall into the in band, avoiding distortion to the ECG signal. Finally, a low resolution ADC samples the full dynamic range and additional signal processing takes place in the digital domain. The sampling rate of the ADCs depends on the ANSI/AAMI requirements as covered in the previous section. With pacemaker detection, the sampling frequency requirement increases substantially; however, the resolution of ADCs is lower compared to DC coupled AFE, and theoretically the 2.5 V full scale ADC voltage requires at least 10 effective bits to achieve 5 µV resolution with a 5 mV input signal.

2.2.2. DC Coupled Systems

Unlike AC coupling, DC coupled systems do not remove the electrode offset and require processing in the digital domain to remove the DC component. Therefore, the total hardware required in the analogue front end is substantially less compared to AC coupled counterparts. Fig. 2.10 shows a conventional DC coupled AFE system architecture for single lead implementation.

The overall architecture consists of instrumentation amplifiers with high CMRR, an anti-aliasing low-pass filter with cut-off frequency defined at 150 Hz and a high resolution ADC to sample the ECG signal. Since the overall system samples the electrode offset in addition to the signal of interest, the overall amplification in the analogue domain is two orders of magnitude less compared to AC coupled systems to avoid saturation.

Given a full-scale ADC voltage of 2.5 V with a 300 mV electrode offset and a gain setting of 5 V/ V to guarantee an unsaturated output, at least 17 effective bits are required to achieve 5 µV resolution. For this reason, these types of implementations often utilise delta sigma ADC structures with oversampling and noise shaping techniques unlike SAR ADC implementations in AC coupling approaches. Therefore, the noise free dynamic range improves compared to the AC coupled systems preserving the signal integrity. The flexibility of signal processing techniques in the digital domain then addresses noise artefacts such as baseline wander, motion artefact, muscle contractions and residual powerline interference.

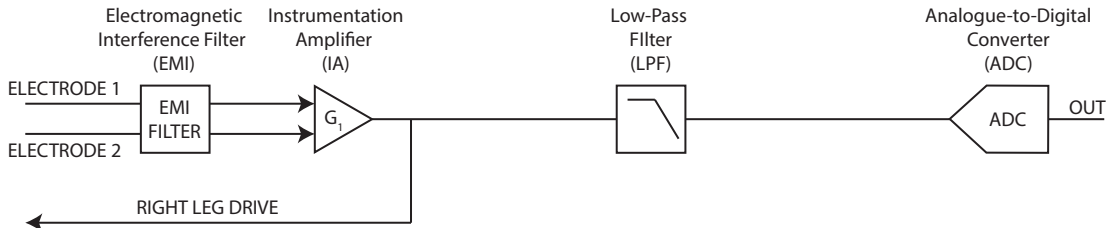


Figure 2.10.: A typical DC coupled front end architecture in ECG systems

2.2.3. Baseline Wander Removal Methods

As covered in Section 2.1.5, baseline wander is the classical disturbance to the isoelectric line mainly caused by respiration. Regardless of the AFE implementations, this type of noise should be eliminated without distorting the ST segment. In this section, the types of baseline wander removal approaches reported in the literature are covered in two main categories namely computational and DSP-Hardware based methods. Computational methods require high number of hardware resources limiting their real-time implementations, whereas hardware-based approaches focus on baseline wander removal with less computational resources with an accuracy trade-off. These methods are discussed in more detail and the reported comparisons of these algorithms found in the literature are also provided.

2.2.3.1. Computational Methods

Empirical Mode Decomposition, (EMD) which was proposed by N.E.Huang in 1998 [49], is an iterative method used to decompose non-linear and non-stationary signals into its components with slowly varying amplitude and phase characteristics. This adaptive approach is used in ECG recordings to estimate and remove band-limited noise interferences.

EMD relies on a fully data-driven mechanism without a priori specification and utilises the decomposed signal functions in estimating the noise interferences. These functions, referred to as intrinsic mode functions (IMFs), are formed through an iterative approach called the sifting process. This process averages the mean of upper and lower envelope functions detected at the local extrema and subtracts the calculated average of these envelope functions from the signal of interest until the sum of difference criterion is smaller than a pre-determined threshold [49]. The result of this process forms the first intrinsic mode function (IMF), which possesses a narrow-band frequency component of the signal. The same process is then repeated on the generated intrinsic mode function until a slowly varying residue remains.

When the process is completed, the signal of interest is decomposed into its high and low frequency components, and the removal of noise interferences can be achieved separately. The high frequency noise components can be filtered out by partial signal reconstruction of the first several IMFs. During such an operation, one should take into account that QRS components lie in the same IMFs with the powerline interference. Therefore, delineation and separation of the QRS complex by using proper windowing techniques are required to preserve the integrity of the complex [50]. Similarly, higher order IMFs refer to the low frequency components and

these are low pass filtered to remove the baseline wander.

Due to the iterative nature of such an approach, the EMD operation is computationally demanding and not suitable for real-time implementations as high order of IMFs are generated by running the sifting process multiple times on recorded data. In addition, the removal of baseline wander is achieved by filters and these can still introduce distortions to the ST segment and limit the implications of such an approach. EMD for baseline wander removal has been previously reported in the literature; however, the acquired results do not follow a certain pattern in all cases.

In the most cited EMD baseline wander removal technique, Blanco-Velasco *et al.* utilise only the first 46000 samples of MIT-BIH Arrhythmia signals (100, 103, 105, 119, 123) sampled at 360 Hz and compare the results to a high-pass Butterworth filter with a cut-off frequency defined at 0.09 Hz [50–52]. In addition, real-time tests involve baseline wander signals with different attenuation levels from the MIT-BIH Noise Stress Database and the signal-to-error ratio (SER) is utilised as the evaluation metric. The reported result of MIT-BIH Arrhythmia signal (100) with 6 dB attenuated baseline wander signal (BW) shows 11.40 dB SER in comparison to 5.22 dB and 6.14 dB SER results of the high-pass filtering and the wavelet approaches respectively. Even though such high-pass filtering is not within the standards and the baseline wander signals have respiration content above these frequencies, ST segment distortion is not mentioned throughout the article. In addition, the first two minutes of these recording are somewhat clear and do not involve step changes or artefacts; therefore true evaluation is unknown.

In the work of Chang, ensemble EMD (EEMD) is tested on baseline wander removal of ECG signals [53]. Reported tests involve 30 minute duration of noise interferences added on the MIT-BIH Arrhythmia Database signals (101, 102, 103 and 104, which are recorded at a sampling rate of 360 Hz). Pre-processing is applied with a band-pass filter cut-off frequencies defined at 1 and 35 Hz and the filtered signal is used as a template. Results of ensemble EMD are compared with a 3^{rd} order Butterworth filter with a 1 Hz cut-off frequency, a 300^{th} order Wiener Filter and a typical EMD approach. The reported errors of MIT-BIH Arrhythmia Database signal 101, show mean square error (MSE) of 0.041, 0.016, 0.001 and 0.0007 with no units mentioned respectively. Also the work of Jenitta *et al.* [54] shows improvements on EMD and EEMD techniques by utilising adaptive filtering of the IMF components.

In another work, EMD and EEMD results on real data and baseline wander show signal to noise ratio (SNR) of 39.2 dB and 36.7 dB respectively [55]. Lastly, an EMD approach with FIR filter implementation showed RMS errors of 55 µV and 34.5 µV for MIT-BIH Arrhythmia Database signal 106 and 111 with no added baseline wander respectively [56].

Other works of EMD based baseline removal methods covered in the literature do not provide a deep analysis to baseline wander removal but rather illustrate plots or mention cross correlation coefficients associated with each removal technique [57–59].

Independent Component Analysis (ICA) is a type of blind source separation method utilised in ECG signals to remove the baseline wander. Similar to the EMD method, ICA does not require a priori knowledge and the signal is separated into its additive subcomponents. ICA methodology is based on the assumption that these subcomponents are non-Gaussian signals and statistically independent from each other [60].

The ICA based researches on biomedical signals including ECGs have been reported in the literature [61,62]. According to these studies, ECGs satisfy some of the ICA conditions, which include superimposing current linearly at the electrodes, negligible time delays and non-Gaussian voltage distribution. A detailed analysis of different types of component analysis to noise and artefact suppression in multichannel ECGs is provided in the literature [63]. In one of the most cited works of ICA in ECG artefact removal, results based on the kurtosis and variance are provided [64]. This study identifies all the components whose Kurtosis modulus is below a certain threshold as continuous noise. While the authors demonstrate the successful removal of noise artefacts, a quantitative analysis is still missing.

Most ICA approaches focus on separation of fetal ECG from the ECG of the mother and a detailed analysis of this method on ECG baseline wander removal is not covered in most of the articles. Nevertheless, SNR improvements with FastICA technique on real data signals acquired from MIT-BIH Arrhythmia Database with artificial baseline wanders generated in MATLAB are reported [65, 66]. Other ICA approaches with plot illustrations of baseline wander removal can be found in the literature [67–71].

Discrete Wavelet Transforms (DWTs) have been used in ECG data compression, baseline drift and powerline interference removal as reported in the literature. These transforms are suitable for transient and non-stationary signals and offer simultaneous interpretation of the signal both in the time and in the frequency domain.

The performance of the wavelet transforms is correlated to the defined mother wavelet in each approach. Rahman *et al.* investigate the effect of 110 mother wavelet functions on ECG baseline wander removal and quantify a compatible mother wavelet function by means of SNR and MSE [72]. Another approach estimates baseline wander by removing the signal mean from the transform space, and setting the low frequency coefficients to zero as these do not appear on the wavelet space [73].

Park *et al.* presents a detailed analysis on ST segment distortion with wavelet approaches [74]. This study utilises "Vaidyanathan - Hoang" wavelets with adaptive filtering and incorporates ST segment distortion defined within standards. MIT-BIH Database tests show that lower ST segment distortion is achieved when compared to a standard high-pass and a general adaptive filter. Similarly, a wavelet adaptive filter structure is utilised to identify ST segment fiducial points in the literature [75].

An example of mean-median filter and discrete wavelet transform aims to remove the baseline wander with a selection of different mother wavelets [76]. In their work, Hao *et al.* compared their results to the EMD technique. Reported MIT-BIH results with artificial baseline wander demonstrate improvements on SNR levels from 11.3 dB to 13.4 dB [76]. However, real data

baseline wander removal response of the proposed method is unknown and not covered in the reported work.

A detailed bionic wavelet analysis shows that bionic wavelet transform and adaptive determination of the centre frequency can be used to decompose ECG signal into its low frequency components. Sayadi *et al.* utilise the first 4096 samples of the MIT-BIH Arrhythmia Database signals sampled at 360 Hz for testing and define "Morlet" wavelet as their mother wavelet [77]. The results are then compared with the noise free signal and improvements on the added white Gaussian noise are quantified. These tests do not involve ambulatory baseline wander signals acquired from the MIT-BIH Noise Stress Database. In addition, a non-local wavelet transform domain filtering is utilised in ECG signal denoising in the literature [78]. MSE and SNR of MIT-BIH Arrhythmia Database signals with added white Gaussian noise are evaluated and results are compared with other methods. Similar to the bionic wavelet analysis, a thorough evaluation of baseline wander removal is not provided with the reported work.

Other works of wavelet based baseline wander removal that do not explicitly mention ST segment error analysis but rather illustrate baseline wander correction, can also be found in the literature [79–83].

Brownian Motion Process, first introduced by Van Ness *et al.*, models the clean ECG signal and the baseline wander as a 1^{st} and a 2^{nd} order fractional Brownian motion (fBm) processes [84, 85]. Eigenvectors of the auto-covariance matrix of clean ECG are then utilised in designing an M-channel uniformly decimated filterbank and the noisy signal is filtered by this filterbank [86]. The same authors also present another approach based on fractional Brownian motion process and achieve baseline wander removal by a projection based operator [87]. Reported results of both methods include 3600 samples of MIT-BIH Arrhythmia records (100, 101, 103 and 115) sampled at 360 Hz and show SNR of 8.0 dB, whereas filterbank, EMD and wavelet analysis achieve SNR levels of 13.5 dB, 4.5 dB and 1.9 dB respectively.

Polynomial Interpolation has been used as a baseline wander removal technique and requires a priori knowledge in baseline estimation, known as the "knots". The order of the polynomial interpolation has been a studied topic and a third order approximation known as the cubic spline interpolation is preferred in the literature [88].

By making use of the previous information of the ECG isoelectric levels at the PR intervals, the baseline drift estimation is generated [89]. The performance of the cubic spline interpolation, however, depends on the detected PR intervals and the heart rate [88]. As the heart rate slows down, the distance between two adjacent PR level increases, which results in degradation of the estimation. In addition, Meyer *et al.* mention that two fiducial points, one during PR-interval, and one during TP-interval are resisted as these two intervals might be at different elevations and heart-generated differences might be removed in such instances.

Froning *et al.* list problems and limitations in regards to cubic spline interpolation based baseline wander estimation and cover current approaches and solutions [90]. The authors mention that linear interpolation works better when baseline wander is "relatively slight" and starts

to degrade with curvatures. The same article also mentions that the accumulation of errors due to arithmetic computation needs to be compensated by a scaling factor in state-space coefficient calculations.

A comparison of polynomial interpolation of synthetic baseline wander with other methods is provided in the work of Gradwohl *et al.* [91]. A single pole (f_c=0.05 Hz), a null phase (f_c=0.75 Hz) and a 6-pole filter (f_c=1.2 Hz) are compared to cubic spline interpolation and the root mean square results show 297.1 µV, 75.4 µV, 58.4 µV and 23.5 µV of deviation respectively.

In addition to that, another cubic spline interpolation approach with real and synthetic data is mentioned in the literature [92]. The reported tests include a single signal acquired from MIT-BIH Fantasia Database with added artificial baseline wander. The analysis is acquired through the evaluated mean square errors and correlation coefficients of the test signal.

Quadratic Variation Reduction (QVR) in ECG baseline wander removal is based on an optimisation problem and the quadratic variation is defined as a constraint [93]. Not many articles exist in this topic; however, Fasano and Villani, provide a detailed analysis and test their approach with synthetic data sampled at 512 Hz, at a heart rate of 75 bpm [93]. The baseline wander is rendered as Gaussian white noise (µ=0, σ^2=6.25) and low pass filtered at 0.8 Hz. Results show better ST segment evaluations when compared to high-pass filter, cubic spline interpolation, median filter, adaptive filter and wavelet adaptive filter. Other works of the same authors in regards to this topic showing results with different datasets are also covered in the literature [94–96].

Multi-scale Mathematical Morphology is a non-linear technique focusing on the shape information of a signal. Basic operators, such as erosion and dilation referred to as grey-scale morphological operators in the signal processing literature, are used in tandem for opening and closing a signal. The main idea is to suppress the impulsive noise by processing the data through a sequence of basic operators. The output of two consecutive operations can be improved by exchanging the operators' order such as "opening-closing-closing-opening" and taking the average of the outputs.

The baseline drift is estimated by background normalisation, which depends on the design of the structuring elements. The peaks of the data are removed by opening it with a structuring element resulting in a pit where the ECG signal is situated. The closing operation is employed with a larger structuring element to remove the pit in order to obtain an estimate of the baseline wander [97]. In other words, structuring elements suppress the peaks and valleys of the ECG, leaving behind the low frequency components like the baseline wander estimate. Finally, a clean ECG signal is obtained by subtracting this estimate. According to the results reported in the work of She *et al.*, MIT-BIH Arrhythmia Database signal (106) and added white noise generated SNR levels of 9.4 dB, 14.4 dB, and 15.0 dB with different thresholding techniques.

In another work, Taouli and Bereksi-Reguig carry out a more detailed analysis. This article investigates the baseline removal on MIT-BIH Arrhythmia Database signals (101, 113 and 209) [98]. The results are evaluated in terms of SNR, RMS, MSE and correlation coefficients

and demonstrate similar results when compared to wavelet approach. MIT-BIH Arrhythmia Database signal 101 result leads to SNR levels of 28.7 dB with the morphological filter whereas the wavelet approach result shows 27.9 dB SNR. It should be noted that these test do not include ambulatory baseline wander acquired from the MIT-BIH Noise Stress Database. Additionally, a similar analysis can be found in the article with artificial baseline wander [99]. The results show that SNR levels degrade significantly and relate to the amplitude of the baseline wander.

In addition, white noise suppression and background normalisation with mathematical morphology are also discussed in the literature [100]. Additional work originated from the same baseline correction technique is covered in the work of Sun *et al.* [101]. In this article, the tests include synthetic and real data acquired from the MIT-BIH Arrhythmia Database. As evaluation metric correct, however, QRS complex detection rates are utilised showing an improvement from 96.7% to 99.4%. Other mathematical morphology methods also can be found in the literature [102, 103].

Median & Moving Average Based Filters are other non-linear and linear techniques that have been utilised in baseline wander removal. Both techniques operate in a similar fashion where new samples get filtered with respect to the median or the average of the corresponding filter's window size respectively.

Empirically it was found in the works of Hao *et al.* that a window size for a mean-median filter that is half of the sampling frequency is more suitable for baseline removal when the output is corrected with a discrete wavelet approach [76]. In another article, Dai and Lian describe a modified moving average filter to avoid distortions to the low frequency content [104]. The cross correlation results show improvement when compared to moving averages and the cross correlation accuracy increases from 0.845 to 0.965.

In the work of Leski and Henzel, mean and standard deviation errors of a moving average filter with a window size of 5000 samples (sampled at 500 Hz) show mean error results of 58 µV with a standard deviation of 42 µV when artificial baseline wander of 48 µV mean and 105 µV standard deviation is added [105]. The tested signals are acquired from CTS-IEC Database and when no artificial baseline wander is added, the maximum distortion is reported as 18.8 µV.

In addition, median filtering when combined with mathematical operators shows an improvement as reported in the work of Verma *et al.* MIT-BIH signals (118 and 119) with added 24 dB attenuated baseline wander signal, show improvements up to 34.4 dB and 24.1 dB.

Infinite Impulse Response (IIR) filters are digital recursive filters with non-linear phase characteristics. Due to the subsequent distortion caused by their phase response, computational methods require zero phase filtering. These type of filters filter the signal in the forward and reverse directions, therefore, require data to be stored in memory. A bilinear transformed filter utilised both ways and RMSD errors are presented in the work of Pottala *et al.* [106]. These RMSD errors are "practically equivalent" to the implemented cubic spline interpolation method. In another work, a two-pole phase compensated filter with a synthetic baseline wander (RMS value equal to 338.6 µV) generates an SNR improvement of 15 dB, whereas a standard 0.5 Hz

single-pole filter improves by 8 dB [107]. In the work of Shusterman *et al.*, selective zero-phase filtering shows improvement in SNR by 13.8 dB (RMS error equal to 46 µV) when simulated baseline wander is utilised [108].

Finite Impulse Response (FIR) filters are used to remove the baseline drift and powerline interference. As these filters have linear phase response, the cut-off frequency requirements defined by the standards are relaxed when compared to non-linear phase response filters. Due to the narrow transition bandwidth, however, FIR filters require high number of taps and high computational complexity.

An FIR study shows removal of baseline wander with reduced number of taps [109, 110]. In this article, the authors define a bandpass filter from 0.8 Hz to 50 Hz used to filter the ECG signal baseline wander and powerline interference with its harmonics. This approach reduces the total number of multiplications per output by a factor of 10. However, results of such a technique are not evaluated with real data and the ripple effect of pass-band and stop-band attenuation might not be sufficient enough in case of baseline wander with high amplitudes. As an improvement to the previous article low-pass FIR filter response is subtracted from the delayed input and the results are illustrated [111].

In the work by Kumar *et al.*, FIR filter approaches with different window sizes are compared with IIR filter responses on real ECG data acquired from MIT-BIH Arrhythmia Database [112]. Other digital FIR filter approaches utilising windowing techniques include [113–116].

Adaptive Filters utilise an optimisation algorithm and adjust the coefficient values of the digital filter according to this optimisation. The least mean squares (LMS) and the recursive least squares (RLS) filters are types of adaptive filters.

In ECG baseline wander removal, adaptive filters with least mean squares algorithm are covered in various articles. Thakor *et al.* describe an adaptive recurrent filter structure with a least mean squares algorithm and investigates baseline wander removal [117]. In the work of Thodetil and Lakshmi, different least mean squares techniques on synthetic ECG and synthetic baseline wander are investigated [118]. Results show SNR ratios of 31.7 dB and 34.2 dB for the LMS and normalised-LMS algorithms respectively. On the other hand, real data signals acquired from MIT-BIH Arrhythmia Database (104 and 105) with no added baseline wander show SNR results of 14.6 dB and 16.4 dB respectively [119]. A detailed analysis on the first 4000 samples of the real data acquired from MIT-BIH Arrhythmia Database records (100, 105, 108, and 228) sampled at 360 Hz with real baseline wander acquired from MIT-BIH Noise Stress Database demonstrates SNR results of 11.1 dB, 12.3 dB, 11.6 dB, 12.9 dB respectively [120]. Similar results can also be found in the work of Paul, and Mythili [121]. In addition to these, other adaptive filter approaches illustrate baseline wander removal in the literature [122–126].

Similarly, recursive least square approaches have been implemented on ECG baseline wander removal. Unlike least means square algorithms, these approaches are more computationally intensive and harder to stabilise. The results, however, converge faster. Real data results of both of RLS and LMS adaptive filters on ECG baseline wander removal are compared in the

literature and as evaluation metrics SNR, RMSE and correlation coefficients of each method are utilised respectively [127–129]. In addition, in the work of Chandrakar SNR improvement of 5.0 dB is reported with real baseline wander signal acquired from the MIT-BIH Noise Stress database on real ECG signals [130].

Kalman Filters are also utilised in ECG baseline wander removal. These types of filters are based on recursive measurements, and generate an estimate of unknown variables. These estimates can be based on various variables and has a better estimated uncertainty than the predicted and the measured states alone. These systems require the last "best guess" in calculating a new state rather than the entire state history. Therefore, storage requirements are minimal.

In ECG baseline removal, this linear estimator has been investigated and results of synthetic ECG are presented [131, 132]. In the work of Sayadi and Shamsollahi, a detailed analysis of MIT-BIH Arryhtmia Database signal portions with real baseline wander acquired from MIT-BIH Noise Stress Database show an SNR improvement of 10.2 dB [133].

2.2.3.2. Hardware-based Approaches

Hardware-based EMD Algorithm is reported in the literature via a digital signal processor (DSP) and a field programmable gate array (FPGA). These processors utilise cubic spline interpolation to define the envelope functions and handle sifting operation on IMFs iteratively [134]. Results are tested on the first 1000 samples of a single MIT-BIH Arrhythmia Database record sampled at 360 Hz with added synthetic baseline wander and a correlation coefficient of 0.9963 is achieved. These results degrade with additional synthetic powerline interference as the number of generated IMFs are limited and blind signal decomposition requires more IMFs to delineate noise interferences. Another implementation of hardware-accelerated EMD approach is covered in the work of Wand *et al.* [135].

AFE Design With DSPs are one of the key hardware approaches for noise removal in addition to the computational approaches described above. IMEC has published hardware-based solutions to remove motion artefacts that are detected via electrode tissue impedance measurements along with other AFE designs [136–141]. In this approach, the electrode impedance is constantly measured independent of the ECG inputs and is fed to a microcontroller where a least mean squares adaptive filter processes the readings. The processed output is then fed back to the analogue front end and the noise estimate is subtracted from the input. As the motion artefact and baseline wander originate from the impedance changes seen by the amplifier, this approach also addresses baseline wander. However, 200 mHz cut-off frequency for the high-pass analogue filter is still above the specified limits for certain ECG segments. Similarly, a long-term baseline wander tracking system is recently published and utilises a microcontroller approach employing MSP430 [142]. This approach reacts fast to offset shifts and report 29 dB SER with synthetic ECG.

Multiplier-free FIR Filters are required in hardware-based baseline wander removal to be implemented in VLSI technology. For this purpose, multiplication free recursive running sum (RRS) filters are utilised with high-pass cut-off frequency defined at 0.5 Hz. In this approach, RRS filters utilise only 37 adders and 2248 delays to obtain a better computational result when compared to an FIR filter with 1149 multipliers and 2296 adders [143]. However, a true analysis and error quantification are lacking in the reported work.

Commercially Available Systems also target baseline wander removal; however, these systems do not provide enough information about their system architecture, accuracy or computational requirements. One such system is the Smartheart, which provides 12-lead telemetry solution [9]. The recorded results are sent to a telemetry centre and the patient is diagnosed by clinicians within three hours; however, neither the utilised baseline wander removal technique nor its accuracy is clear. Similarly, Sensium provides real-time medical care solutions by recording patients' vital signs and transmits these signs wirelessly [10]. Alivecor, on the other hand, generates ECG results in 30 seconds with "FDA-cleared machine learning algorithms" [144]. There are of course other available systems on the market; nevertheless, as the utilised methods and their accuracy remain unknown, they do not provide any meaningful additional insight.

2.2.4. Comparison of Algorithms

Different algorithmic approaches in baseline wander removal were presented previously. The evaluation metrics utilised in each method, however, vary from one another. Some approaches only utilise synthetic signals, whereas others use sections of real data from various databases in addition to varying characteristics of the added baseline wander. Moreover, evaluated results in some are obtained by employing original databases as "clean" ECG signals, whereas some approaches utilise the filtered versions instead. Some approaches show only SNR improvements;

Table 2.3.: Comparison of algorithms - ESC ST-T Database ST segment

Method	Mean deviation from the isoelectric line in µV	Median deviation from the isoelectric line in µV
Cubic Spline Int.	85.4	53.6
Linear Spline Int.	77.9	55.0
MF	86.1	55.9
AF	78.9	56.9
WAF	67.6	42.6
FIR HPF	73.3	50.9
EMD	76.7	54.1
QVR	64.4	32.1

Table 2.4.: Comparison of algorithms - MIT-BIH Arrhythmia Database ST segment

Method	Mean deviation from the isoelectric line in µV	Standard deviation from the isoelectric line in µV
HPF	9.7	45.0
MF	6.7	28.8
Adaptive Filter	7.9	32.3
WAF	5.8	25.9
QVR	4.3	20.7

however, the original signal is usually corrupted with powerline interference and these improvements do not yield a true evaluation in baseline wander removal. Therefore, a true comparison of utilising the results of each work is hard to evaluate.

Based on the diversity of evaluation metrics, algorithm comparison articles are investigated in the literature. In the work of Afsar *et al.*, different algorithms are tested with the European Society of Cardilogy (ESC) ST-T Database, which include two hours of two-channel ECG data sampled at 250 Hz. Table 2.3 presents the amount of mean and median of maximum distortion of these methods evaluated from this database. Later, Fasano *et al.* added quadratic variation reduction results to this table [96].

In another work, Fasano and Villani utilises MIT-BIH Arrhythmia Database with eight non-overlapping realisations of added baseline wander from MIT-BIH Noise Stress Database added to each channel of recording 119 [94]. However, these non-overlapping realisations of baseline wander are not stated clearly and reported results are shown in Table 2.4

2.3. Summary

A comprehensive literature review shows that over the years new computational methods for ECG baseline wander removal are being developed. Due to the high efficiency requirements, however, challenges of ECG signal processing to facilitate ambulatory applications still remain to be investigated.

New approaches are being considered to find innovative and comprehensive solutions to address the removal of noise interferences while preserving the signal integrity. These approaches, especially the ones that base their methodology on ECG morphology, require a thorough understanding of heart activity and its dynamics. Of course, the studies presented here by no means cover the full spectrum of work in relation to ECGs but rather provide a background for the reader to comprehend the methods developed in the following chapters as well as defining the true nature of problems associated with noise interferences.

The methods that can be found in the literature in regards to baseline wander removal are reviewed in detail and where available statistical measures associated with each method are

provided. As reported, baseline removal methods are computationally extensive with limited hardware-based approaches existing. The lack of a standardised measurement process for each algorithm, however, challenges a thorough evaluation of each algorithm.

Advances of the methods developed years ago, remain limited and no reported study investigates whether these methods can be improved further. The feature of interest that is covered throughout the thesis will be focused via a new interpolation-based baseline wander estimation originated from cubic spline interpolation. Due to its non-filter based removal characteristics, distortions to the signal of interest, especially to the ST segment, are limited and carry a significant importance in the context of myocardial infarctions.

Chapter 3

An Isoelectric Point Based ECG Baseline Removal Algorithm

ECG's non-invasiveness coupled with the growing trend in wearable technology has given ambulatory systems a resurgence in daily healthcare applications. However, in ambulatory monitoring there are still critical challenges for accuracy, noise and artefact removal without compromising the clinical validity of the ECG. Therefore, one needs to understand ECG morphology thoroughly and know how to address each noise interference in the signal processing chain prior to system realisation.

The main motivation in this chapter is to address the baseline wander in the presence of other noise artefacts in a resource efficient manner and identify/characterise all the system parameters in ECG signal processing. The comprehensive literature review in the previous chapter showed that ambulatory system designs rely on high-pass analogue filters to remove baseline wander. These approaches introduce unacceptable distortion to the signal of interest due to the non-linear phase responses of these filters.

Noise artefact detection accuracy depends on the interference characteristics and the robustness of the methods used. However, the sensitivity of a specific feature to noise or to distortion in ECGs varies from feature to feature. One such feature in particular that is highly susceptible to noise and distortion is the ST segment. Within this context, it is the aim of this chapter to develop a novel method for ECG baseline drift removal while preserving the integrity of the ST segment.

The remainder of this chapter is organised as follows: Section 3.1 lists the main objectives, Section 3.2 provides a brief background on noise artefacts present in ECG recordings and describes the challenges; Section 3.3 introduces a new baseline wander removal algorithm and discusses system methodology; Section 3.4 details the datasets and evaluation parameters used in testing; Section 3.5 focuses on the computationally efficient parameter selection; Section 3.6 presents and discusses the results with complex algorithms; while Section 3.7 concludes the chapter.

3.1. Objectives

This chapter focuses on a novel baseline wander detection algorithm that is suitable for ambulatory systems. The main objectives of this chapter can be summarised as follows:

- **Distortion:** Baseline wander is required to be eliminated in ECG recordings to acquire a "clean" reading on ECG signals; however, utilising high-pass analogue filters like most conventional systems do, distorts the signal integrity. The developed algorithm is therefore required to detect and accurately remove the baseline wander in the presence of other noise interferences while preserving the ECG signal integrity.
- **Adaptability to Biological Signals:** ECG characteristics vary not only from person to person but also from beat to beat as there are many internal and external factors that can cause heart rates to fluctuate. These can depend on certain conditions such as illnesses, diseases or can be experienced during emotional circumstances or physical exertions [145]. Therefore, the developed algorithm is required to adapt to changing ECG dynamics and evaluate baseline wander accurately.
- **Computational Complexity:** Clinically valid systems utilise computationally complex algorithms requiring high number of multiplication operations as well as extensive amounts of data storage space. These techniques require windowing approaches and large chunks of data storage to obtain time-frequency based analysis [146]. Due to these requirements, they are not viable in ambulatory design and therefore, developed algorithm has to be efficient and require low computational complexity.

3.2. Background

3.2.1. Noise Interferences

Noise and interference pose significant challenges to the signal processing chain in an ECG system, particularly when they have spectral content within the ECG bandwidth (0.05 - 150 Hz) and are comparable in amplitude (2 - 3 mV). Typical noise interferences that fall into this category but not limited to include baseline drift, powerline interference, muscular contractions, and motion artefacts as shown in Fig 3.1. These noise sources are present during baseline detection and their origins and characteristics are essential in dealing with baseline drift. A detailed description of each noise interference is provided in the previous chapter.

Briefly, these noise sources can be summarised as follows: (1) *Baseline Drift* (often referred to as baseline wander) is the result of the electrode skin impedance changes due to respiration and can be as much as 15% of the peak-to-peak (p-p) ECG amplitude; (2) *Motion Artefacts* occur due to impedance changes associated with movement/vibrations and last for approximately 500 ms with amplitudes up to 5 times the p-p ECG signal [40]; (3) *Muscle Contractions (EMG)* are related to skeletal muscle movement with a range of 50 µV - 2 mV and frequency components from DC to 10 kHz [40]; (4) *Powerline Interference* occurs due to the capacitive coupling from

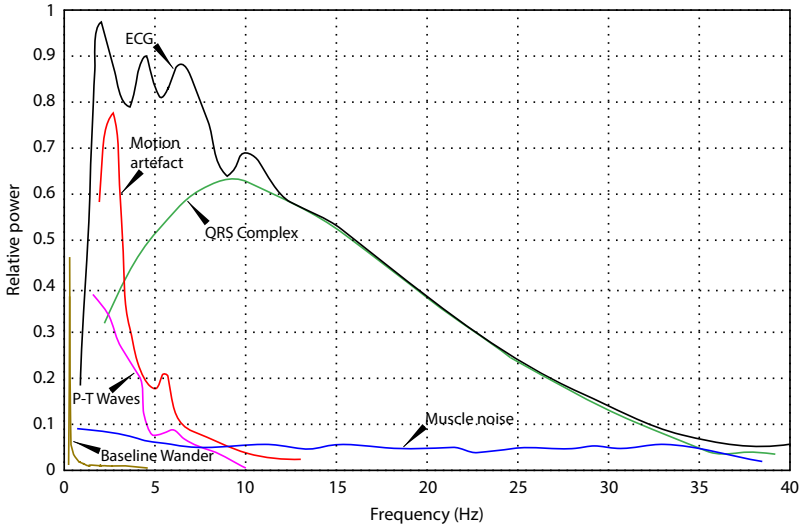


Figure 3.1.: Power spectra of ECG components. Redrawn from [147]

the mains supply to the patients body with its fundamental frequency depending on the region (50/60 Hz).

The focus of this work is the removal of baseline wander, which poses one of the main challenges for ambulatory ECG systems. As motion artefacts also originate from impedance changes seen by the amplifier, the focus of this work also applies to removal of these interferences as long as the signal of interest is not corrupted. In addition, other noise sources can still be present during baseline detection and degrade system performance.

3.2.2. Isoelectric Point Definition

In baseline wander estimation, fiducial points will be forming the basis of detection algorithm and will be referred to as *J1, J2* and *J3* points throughout this chapter. These points are isoelectric landmarks on the ECG complex with slight elevation differences that are detected at different time intervals. For simplicity, these points are considered as isoelectric and will be described in more detail later on.

A typical ECG pattern showing isoelectric/fiducial points *J1, J2* and *J3* along with key futures of each segment/interval is presented in Fig. 3.2. As can be seen, *J1* point is located after the P wave (referred to as P offset in the literature) within the PR segment; *J2* point is detected after the S point (referred to as S offset in the literature) within the ST segment; and *J3* point is situated after the T wave (referred to as T offset in the literature). Here, intervals include at least a wave in their representation, whereas segments are denoted between the onsets and the offsets of the particular waves. These features are utilised to deduce physiological parameters such as R-R interval (heart rate), QRS duration (ventricular depolarisation), ST segment activity and many more.

The physiological explanations of the ECG waveform along with its nominal characteristics are covered in Section 2.1.1, whereas the table in Section 2.1.2 shows the typical durations of each wave, interval and segment.

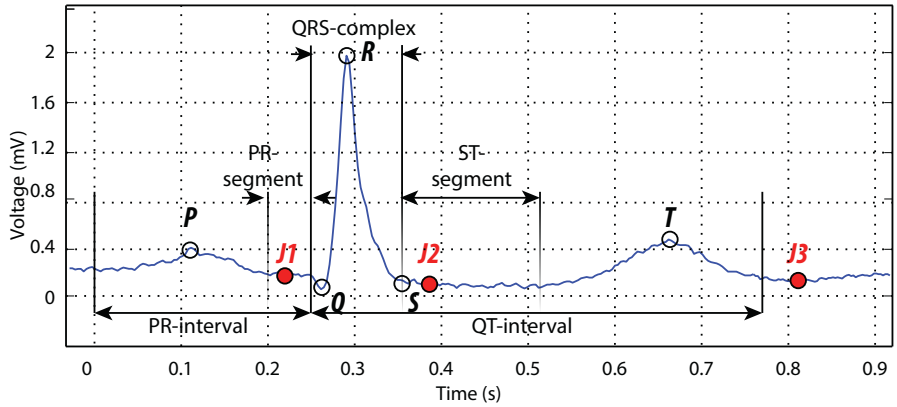


Figure 3.2.: Typical ECG waveform showing key features and isoelectric/fiducial points (J1-3), used here to track the baseline wander

3.2.3. Challenges

Developing a new baseline wander algorithm has its challenges; deciding on which physiological parameters to focus on; how to implement its structure; and computational complexity requirements while preserving the signal integrity. Methods that are clinically valid often demand a high number of computational resources, whereas approaches suitable for ambulatory systems often distort the signal of interest. To address the requirements of both systems, a complete list of challenges can be summarised as follows:

- The designed system has to maintain certain standards and improve accuracy when compared to existing ambulatory systems. Therefore, in baseline detection multiple approaches (i.e. structural, iterative) have to be investigated thoroughly to achieve the best possible accuracy results.

- The developed algorithm is required to estimate and remove the baseline drift without distorting the ECG signal as defined by the standards of the American Heart Association (AHA) and International Electrotechnical Commission (IEC) [17]. It is stated in the literature that a maximum of 100 µV distortion is allowed at the ST segment [18–20]. Any algorithm addressing baseline wander detection is required to follow these constraints and preserve signal integrity.

- The method must adapt to ECG signal dynamics. In other words, it should not be affected by changing signal characteristics and/or physiological disturbances and preserve the ECG signal integrity to its maximum.

- Computationally, processing should be kept to a minimum without requiring high number of operations (i.e. multiplication). In addition, data storage should be kept light as real-time applications are targeted.

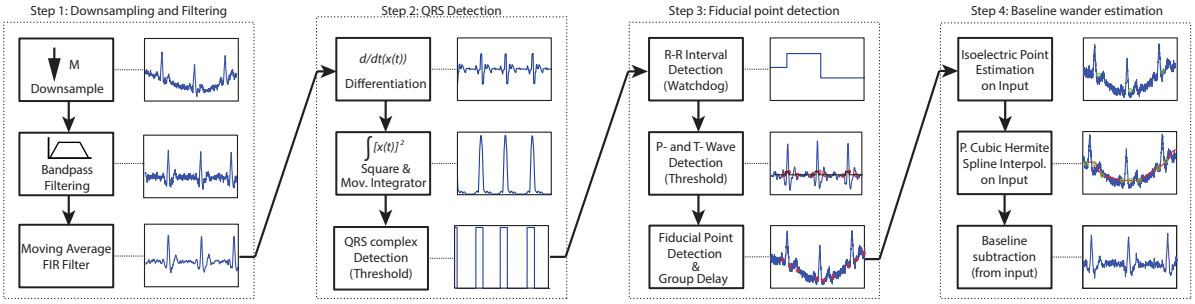


Figure 3.3.: Baseline wander algorithm showing indicative signals through the datapath.

3.3. Real-time ECG Baseline Removal Algorithm

Ambulatory electrocardiograms can be used to monitor patients for myocardial ischaemia; however, often the recordings are contaminated by noise interference that needs to be eliminated. A novel method to address ECG baseline drift removal is developed and presented in this section. The proposed algorithm utilises multiple tested structures to increase accuracy, aims for a computationally efficient real-time approach to decrease power consumption, and achieves accurate baseline wander estimation while preserving the ST segment integrity.

3.3.1. Methodology

The algorithm crudely removes noise artefacts facilitating P, T wave and QRS complex detection, locates three distinctive isoelectric fiducial points, and estimates the overall baseline drift by interpolation. The entire signal chain for this baseline wander estimation process is shown in Fig. 3.3. The proposed algorithm consists of four main stages: (1) downsampling and filtering (to crudely filter baseline drift and EMG interference only for fiducial point detection); (2) QRS detection (to detect QRS complexes based on non-linear thresholding); (3) fiducial point detection (to locate distinctive isoelectric points); and (4) baseline wander estimation & subtraction (to remove estimated drift on the raw ECG data). Each key stage is described in the following sections.

3.3.2. Downsampling & Filtering

The first stage aims to crudely remove noise artefacts (with limited distortion) such that fiducial points can be detected accurately in the following stages. To achieve this with low computational complexity, the signal is first downsampled and then filtered by multiple sub-blocks.

Downsampling the input signal by a factor of M relaxes the transition bandwidth requirements of bandpass filtering, hence reduces the number of operations required by the overall system and the total number of processed samples. Following this, the noise interference is addressed and eliminated coarsely: (1) A high-pass and a low-pass filter with cut-off frequencies f_L and f_H rejects the electrode offset, baseline wander and high frequency content respectively; and (2) a moving average filter suppresses muscle artefacts so that next stages can detect fiducial points.

3.3.3. QRS Detection

The QRS detection stage utilises a (2*N+1) point derivative transfer function which is adapted from the Pan & Tompkins method and has the form in the Z-domain as in Eq. 3.1, where G denotes the gain, N defines the window size and, a_n denotes the coefficients of the transfer function [148–150]. The transfer function is then used to derive the 5-point difference equation as in Eq. 3.2. This way the derivative of the overall function (DC to 30 Hz range) is approximated close to the ideal derivative calculations facilitating real-time operation.

$$H(z) = G * \sum_{n=1}^{N} \frac{a_n}{2} * (z^n - z^{-n}) \tag{3.1}$$

$$y(nT) \approx G * \frac{a_2 * x(nT) + a_1 * x(nT - T) - a_1 * x(nT - 3T) - a_2 * x(nT - 4T)}{2} \tag{3.2}$$

Following the derivative calculations, the output is then squared and passed through a moving window integrator as in Eq. 3.3. In this equation, K denotes the width of the moving integrator window and is determined based on three factors such as the duration of widest QRS complex, the downsampling rate, M, and the sampling frequency, f_s.

$$y(nT) = \frac{1}{K} \sum_{i=0}^{K-1} x\left(nT - iT\right)^2 \tag{3.3}$$

Finally, an adaptive threshold is compared to the output of the integrator to locate QRS complexes. This adaptive threshold is updated once per heart beat as a function of the previous threshold value and the new detected R peak value based on the relationship in Eq. 3.4, where the coefficient values, a and b, are determined empirically.

$$\text{Threshold(n)} = a \cdot \text{Threshold(n-1)} + b \cdot \text{New R Peak(n)} \tag{3.4}$$

3.3.4. Fiducial Point Detection

Once the QRS complex is detected, the algorithm introduces a flagging system to locate P and T waves. This operates as follows: QRS detection raises the QRS flag, initiating the T wave search using the reduced QRS threshold value. After this threshold crossing detection, T wave is located when the derivative changes sign. Similarly, T wave detection raises the T flag and P wave search takes place with a further decrease of the T threshold value, whereafter the algorithm again searches for a derivative sign change. Fig. 3.4(c) shows in detail the threshold technique used to detect P, T waves and QRS complexes.

This way all three waves on a normal ECG rhythm are located. To detect the isoelectric fiducial points, *J1, J2* and *J3*, delays are introduced after each detection based on nominal ECG temporal characteristics and the derivative of the signal is checked if equal to zero. This assumes a P wave duration = 80 ms, PR segment = 50 to 120 ms, QRS complex duration =

120 ms, ST segment = 80 to 120 ms and T wave duration = 160 ms [28].

As the fiducial point search relies on the detection of the QRS complexes, T and P wave searches are not conducted until a J2 fiducial point has been located. If one or both of these waves are missing, the algorithm aligns itself to the QRS complexes and continues on generating the baseline wander with less fiducial points. Additionally, in the event of missing QRS complexes or no detections the system re-initialises itself to recover. On the other hand, in the event of large EMG signals and multiple threshold crossings the algorithm only allows a single fiducial point (*J1*). Therefore, the fiducial point locations temporarily stored in the buffer up to the QRS complex are discarded once a (*J1*) location is validated with an R peak detection. Similarly, multiple (*J3*) threshold crossings are discarded once (*J2*) fiducial point is detected and (*J3*) fiducial point is accepted only when it is within nominal ECG characteristics.

A recovery operation is triggered when a large amplitude step change or motion artefact increases the new threshold value such that the next QRS complex never crosses the new threshold and the system needs to be recovered to proceed. Therefore, in all cases this operation has been implemented as a function of heart rate corresponding to 40 bpm in all cases since lower rates correspond to absolute bradycardia [28].

3.3.5. Baseline Wander Estimation

Even though *J1*, *J2* and *J3* fiducial points are referred to as isoelectric up to this point for simplicity, these fiducial points are not necessarily at the same elevation due to patient-specific heart muscle contractions, relaxations and also certain conditions [88]. These electrical differ-

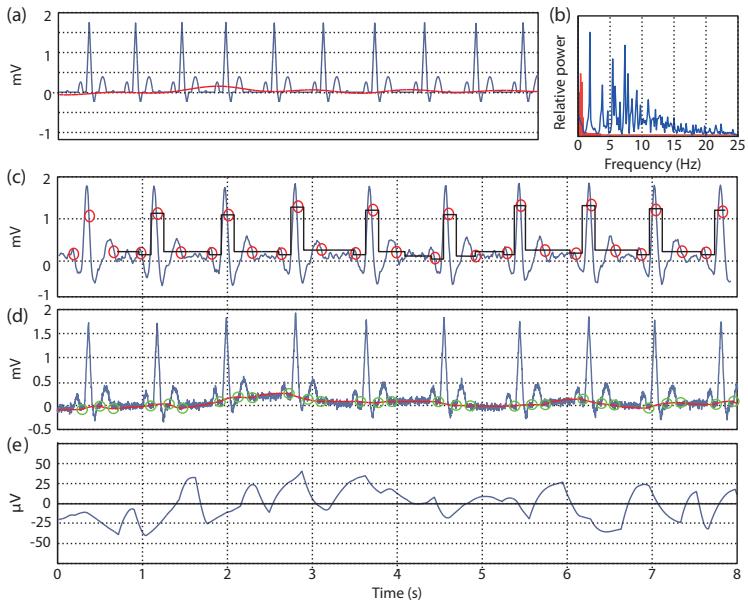


Figure 3.4.: Time domain operation of the proposed algorithm using an 8 second synthetic ECG with added noise artefacts. Shown are: (a) synthetic ECG/baseline drift; (b) relative power spectrum of synthetic ECG/baseline; (c) emphasised signal for R, P and T wave detection; (d) fiducial point detection; (e) baseline wander error

ences between detected fiducial points *J1, J2* and *J3*, have to be compensated to estimate more accurate baseline wander. Therefore, a temporary variable stores these differences (J1-J2 & J2-J3) and the isoelectric point estimation sub-block introduces these errors and interpolation occurs on the corrected fiducial point.

Finally, the baseline estimation is generated by interpolating fiducial points *J1, J2* and *J3* using one of three interpolation methods. These include: piecewise cubic hermite (PCHIP), cubic spline and linear interpolation. Fig. 3.4(d) and (e) show baseline wander estimation and error analysis on a 8 second synthetic ECG signal.

3.4. ECG Data & Evaluation Metrics

The overall system is tested and validated using both synthetic and real ECG signals in MATLAB R2015b platform. Synthetic data sets are utilised for quantifying the effect of individual design parameters (e.g. filter frequencies) on the overall system performance, since the ground truth is known. Once determined, real data sets from the MIT-BIH Databases are used for determining the overall system performance. Both of these data sets are described in this section.

3.4.1. Synthetic Data For System Design

Synthetic data are referred to as a combination of simulated ECG signals, generated using Fourier series approximations [151], whereas two real baseline wander recordings, namely BWM1 and BWM2, are extracted from the Noise Stress Database in Physionet [152]:

- *The simulated ECG* signals are Lead-II representation of regular heart beats in a 12-Lead ECG system. As the ECG signal is quasiperiodic and satisfies Dirichlet's conditions, Fourier series approximation can be utilised to express ECG signal. By decomposing the overall signal into smaller segments and defining optional parameters, ECG representation can be customised. These optional parameters include RR interval, P, R, and T wave durations as well as their amplitudes. For this study, a 30-minute long segment of data is utilised (650,000 samples at 360Hz) with a random variation (up to 10% from the previous heart beat) in all parameters. The P, R and T wave durations with RR intervals are capped to avoid waves overlapping in time and deform other segments.

- *The baseline wander* signals provided in the MIT-BIH Noise Stress Database, are recorded in ambulatory settings with a gain of 200 V/V, under various extreme conditions. Usually, baseline wander is expected to be 15 % of the peak to peak ECG amplitude and can be modelled as a sinusoid [40]. However, in these datasets baseline wander generally reaches up to 100 % of the peak to peak ECG amplitude and higher in some instances. It should be noted that the presence of white Gaussian noise in addition to the baseline wander degrades the baseline estimation since the noise floor here is defined by the white noise. To avoid this, when investigating design parameters, these signals are passed through a 16^{th} order moving average filter.

3.4.2. Test Data For System Evaluation

After design parameter selection, the algorithm is tested on both synthetic and real data from MIT-BIH Databases:

- Synthetic data are generated as described previously, with an attenuated baseline wander in amplitude (from MIT-BIH Noise Stress Database) at ratios of 0, 6, 12, 18 and 24 dB [152]. Therefore, we are able to assess the accuracy of the algorithm at different levels of baseline wander.
- The algorithm is then tested on 12 half hour long recordings from the MIT-BIH Arrhythmia Database. These are each sampled at 360 Hz with 11-bit resolution over a ± 10 mV range and the datasets are annotated by at least two cardiologists.
- Real baseline wander signals (from the MIT-BIH Noise Stress Database) are combined with real ECG recordings from MIT-BIH Arrhythmia Database (specifically, datasets 100 and 101) with signal to baseline wander ratios of the amplitude at 0, 6, 12, 18 and 24 dB.

3.4.3. Evaluation Metrics

3.4.3.1. Synthetic Data

To evaluate the performance of the algorithm, the clean (synthetic) ECG signal is annotated. The second derivative of the synthetic ECG is squared and a threshold applied to determine the fiducial points and separate the signal into sections.

Once the segments are separated, the estimated baseline wander is compared with the real baseline wander for every RR interval and ST segment at the annotated locations. The root mean square deviation (RMSD) of the real, y_{r_i}, and the estimated, y_{e_i}, baseline wander over a defined duration, $1:n$, is evaluated as the accuracy metric as in Eq. 3.5.

$$RMSD = <\epsilon^2> = \sqrt{\frac{\Sigma_{i=1}^{n}(y_{r_i} - y_{e_i})^2}{n}} \tag{3.5}$$

3.4.3.2. Real Data

Similarly, for the real ECG signals annotations from the MIT-BIH Databases are read with the code (*rdann*) provided in the Waveform Database (WFDB) toolbox for MATLAB [153]. RMSD calculations are done by estimating the baseline with a high order low-pass equiripple FIR filter with a transition bandwidth defined at 0.55 Hz to 0.67 Hz (f_s=360 Hz), 0.01 dB passband and 80 dB stopband attenuation. MATLAB *fdatool* defines the minimum order of such filter as 12218^{th} order filter. This way, even though the ground truth is not known, a good and accurate approximation of baseline wander algorithm is targeted. Finally, the cross-correlation matrix of the estimated and the real baseline wander are determined.

3.5. Design & Implementation

In this section, the sensitivity of key design parameters of the baseline wander estimation algorithm is investigated. Each parameter is then determined based on the overall accuracy and its computational requirement.

3.5.1. Downsampling Rate

In digital signal processing, decimation is the processes of reducing the sampling rate of a signal. This approach requires low-pass filtering to mitigate aliasing [154]. Downsampling on the other hand is a more specific term that does not require anti-aliasing filters and only focuses on reducing the sampling rate. Due to its nature, such an approach is susceptible to distortion as noise gets folded into in-band, therefore limiting its applications.

Fig. 3.5 illustrates fundamental concepts of decimation process in detail. Here, $x[n]$ is defined as a sampled representation of a continuous function $x(t)$ at a sampling frequency f_s. To decimate the signal without introducing aliasing, a low-pass filter with a cut-off frequency, f_c, filters the signal and the bandwidth, B_c, satisfies the inequality as defined in Eq. 3.6.

$$B_c < \frac{1}{M} * \frac{f_s}{2} \tag{3.6}$$

In ECG applications, signal content below 45 Hz sampled at 360 Hz can be preserved with a downsampling rate of 4. Even though increasing the downsampling rate, M, reduces the number of operations by $M - 1$ per output, the trade-off is the accuracy degradation in fiducial point detection. This is due to shrinking window sizes of ECG characteristics with downsampling. The maximum window size for QRS complexes of MIT-BIH Arrhythmia Database lasts for 44

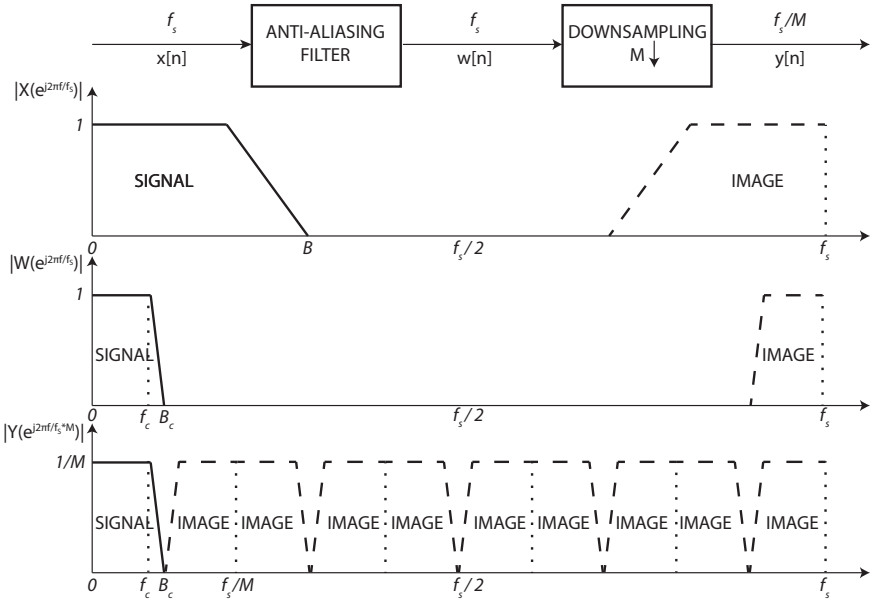


Figure 3.5.: Illustration of decimation with anti-aliasing filter

samples [155], and typically ranges between 22 to 36 samples [28] when sampled at 360 Hz. Therefore, increasing the downsampling rate reduces total number of samples and requires precise thresholding to avoid accuracy degradation.

In the baseline estimation approach presented here, the main aim is to reduce the cost of processing while still being able to detect fiducial points. As ECG characteristics in Fig. 3.2 demonstrate most of the signal content lies at low frequencies and diminishes at higher frequencies, downsampling can still be achieved with small aliasing effects without utilising an anti-alias filter. This preserves computational resources as the decimation filter implementation operating at the sampling frequency, f_s, is discarded.

A thorough quantification of preserving computational resources can be identified with MATLAB simulations. The MATLAB *fdatool* requires a minimum order of 153 to design an anti-aliasing low-pass equiripple FIR filter implementation with a transition bandwidth of 55 to 60 Hz and a stopband/passband attenuation of 20 dB/0.01 dB respectively. The filter order reduces down to 77 when the transition bandwidth is defined from 50 to 60 Hz. As the total number of multiplication operations are determined by the total number of taps defined for the anti-aliasing filter operating at the sampling frequency, f_s, such filters increase the computational load extensively. On the other hand, even though IIR filter implementations for the same transition bandwidths, reduce the filter order required, non-linear phase responses distort the signal quality. Similar to the FIR filter design, low-pass Butterworth IIR filter requires a minimum order of 52 and 25 for the same transition bandwidths respectively and the phase response is almost linear up to 40 Hz in both cases. Increasing the downsampling rate and defining the filter transition bandwidth close to 10 - 30 Hz range, however, affect the signal quality and degrades the system performance. This is due to the non-linear phase distortion of the filters and most of the ECG signal power being defined within that range as can be seen in Fig. 3.1.

With no significant improvements on ECG fiducial point detection, tests have been carried without introducing anti-aliasing filters. Using the test data sets described in Section 3.4.1, downsampling rates are chosen as, $M = 1, 2, 3, 4, 6$ and 8. Fig. 3.6 shows the overall impact error by varying the downsampling rate. As described in Section 3.4.1, the white noise over the BWM1 data defines the noise floor and thus acts as the limiting factor. To present the baseline wander estimation accurately, a moving average filtered version of this BWM1 noise has also been tested and presented with the blue bars along with the unfiltered version of the BWM1.mat file. As can be observed from the plots, for all the plots with white noise removed, the RMSD errors decrease substantially. It can also be observed that the errors (for each downsampling rate) are less than 100 µV, even when using $M = 4$. (For $M \leq 4$, downsampling rate does not affect noise; For $M \geq 6$ there is significant SNR degradation is observed.)

Downsampling also impacts subsequent filtering stages by reducing complexity significantly. For example, a 12^{th} order moving average (MA) filter would be required on the original signal, whereas a 3^{rd} order MA filter is sufficient for a downsampling rate of 4. This also applies to the bandpass filtering. For the following sections, downsampling rate has been fixed to 3 as this provides a good trade off between complexity and accuracy for the heart beats with short ECG intervals.

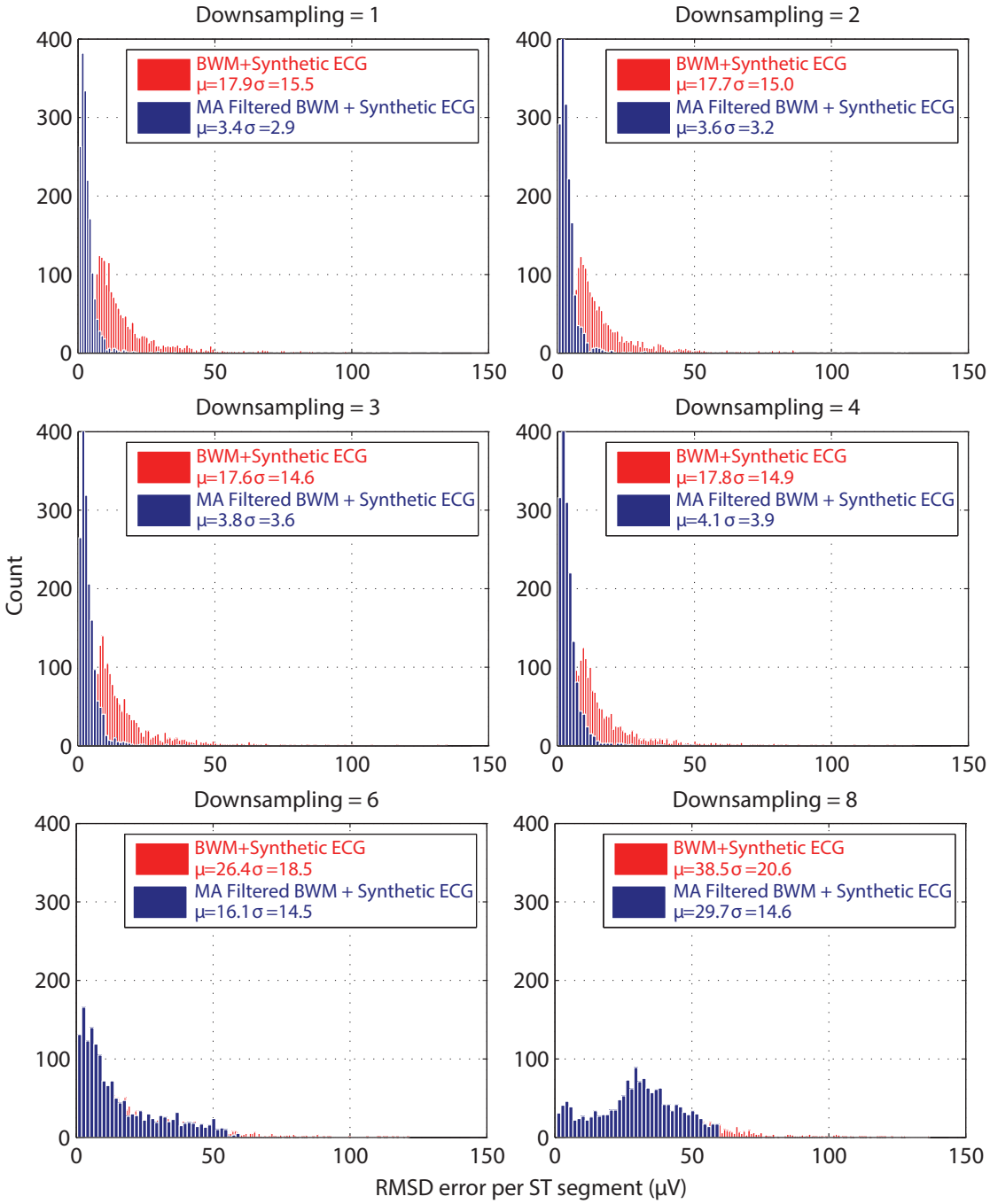


Figure 3.6.: RMSD error vs downsampling rate over 1700 synthetic ST segments using MIT-BIH Noise Stress Database baseline wander (BWM1.mat) and moving average (16^{th} order) filtered version of the baseline wander

3.5.2. Filtering

Once downsampling reduces the sampling frequency to $\frac{f_s}{M}$, filtering stage crudely removes noise artefacts at this rate. As the transition bandwidth requirements are relaxed, noise artefact removal is achieved with minimal computational requirements.

3.5.2.1. High-pass Filtering

The high-pass filter removes the baseline drift for detecting the QRS complex. Typically, FIR filters are preferable because of their linear phase characteristics [156]. However, to reduce the computational complexity of the system, IIR filter responses are investigated. The filter coefficients are generated by matching the filter specifications to the stop-band frequency and varying this parameter as this has the most impact on the signal components (see Fig. 3.1). Note that IIR filtering will introduce distortions in the ST segment, but here the goal is only to detect the fiducial points accurately to determine the baseline wander. The ST segment recorded by the system is not affected by this filtering.

As can be seen in Fig. 3.7, IIR filters here can achieve a similar accuracy to FIR filters (e.g as in [156]) without requiring a high number of coefficients. All the IIR filters can be implemented using only 10 coefficients (3^{rd} order composed of 2 sections) whereas an FIR filter with a transition bandwidth defined as 0.05 - 4 Hz requires 50 coefficients at the downsampled rate. This high number of FIR coefficients not only increases the computational complexity but also requires the original signal to be delayed due to the large group delay. Therefore, IIR filters offer a better design choice for real-time implementation where the phase can be compromised.

As the high-pass frequency is increased, the IIR filter accuracy degrades (see Fig. 3.7). This is expected due to low frequency components of P waves. On the other hand, a cut-off frequency below 1 Hz does not sufficiently remove noise artefacts resulting in an inefficient baseline estimation, hence compromising accuracy (ST segment distortion). Fig. 3.7 shows that Butterworth and Chebyschev2 filters show the best results with the pass-band defined at 1 to 2 Hz. A Butterworth filter with $F_c = 1.5$ Hz, 20 dB stop-band attenuation and 0.5 dB pass-band ripple shows the least RMSD error among all the filters compared and implemented as the filter type.

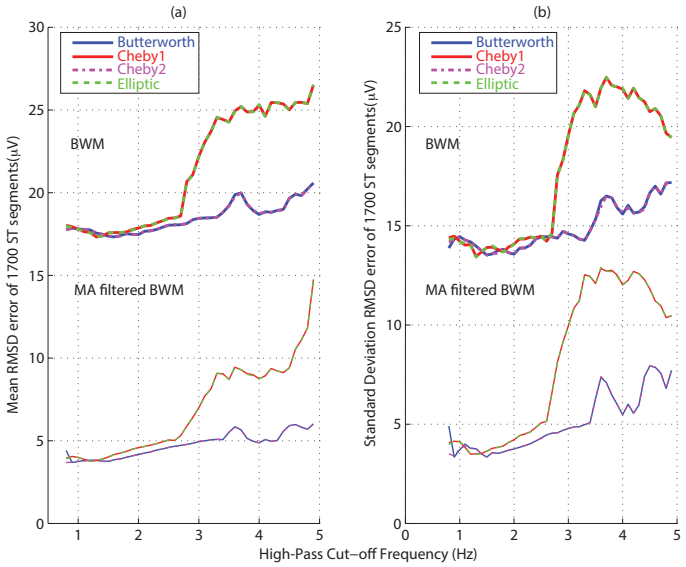


Figure 3.7.: High-pass filter (matched passband) accuracy plots on 1700 synthetic ST segments with baseline wander (BWM1.mat / (16^{th} order) MA filtered): (a) Mean RMSD error vs cut-off frequency (b) Standard Deviation RMSD error vs cut-off frequency

3.5.2.2. Low-pass Filtering

As discussed in Section 3.3, low-pass filtering stage is implemented after the downsampling stage to relax computational requirements. Consequently, images of the harmonics of the powerline interference alias into the pass-band as in Eq. 3.7. Theoretically, a downsampling rate of 3 generates aliases either at 20 Hz ± 1Hz and 30 Hz ± 1Hz (2^{nd} and 3^{rd} harmonic of 50 Hz powerline interference) or at 0 Hz ± 1Hz (2^{nd} harmonic of 60 Hz powerline interference) at a sampling rate of 360 Hz. Low frequency content is filtered by the high-pass filter stage whereas 20 Hz and 30 Hz components are marginal when compared to QRS complexes and do not affect QRS detection.

$$f_{alias} \stackrel{d}{=} |f - N * f_s| \; where \; N = 1, 2, 3...\infty \tag{3.7}$$

Low-pass filtering introduced in this stage removes the fundamental tone of the residual powerline interference and the high frequency noise in fiducial point detection. This way, multiple threshold crossings are avoided to a certain extent; however, these noise interferences are still present in the original signal and accuracy improvement is subject to the defined noise floor. The filter operates at the new sampling frequency defined by the downsampling rate, $\frac{f_s}{M}$. Similar to the high-pass filtering, IIR filter responses are investigated to reduce the computational complexity. The main difference in such implementation, however, is the matching of filter coefficients to the stop-band frequency as the signal of interest here is the QRS complex and filtering of this segment degrades the system performance. This can be seen in Fig. 3.8, whereas at higher cut-off frequencies the filter responses are similar as the residual powerline interference defines the noise floor. Therefore, Butterworth filter with F_c = 35 Hz cut-off frequency 20 dB stop-band attenuation and 0.3 dB pass-band ripple characteristics is selected as the filter type.

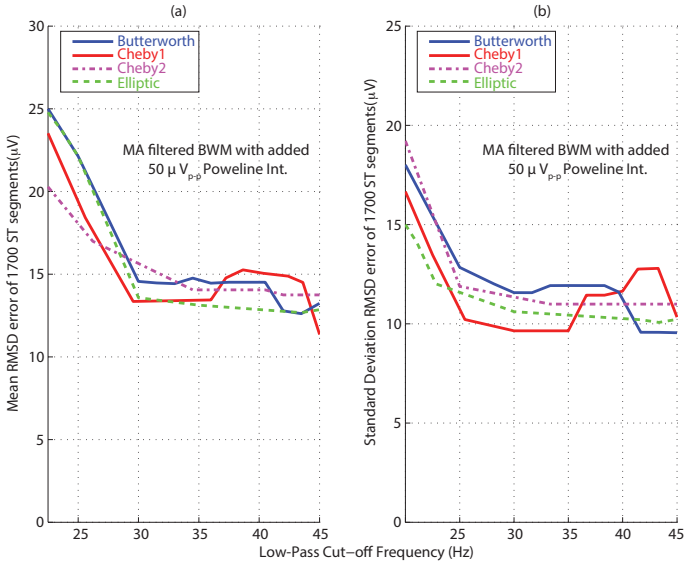


Figure 3.8.: Low-pass filter (matched stopband) accuracy plots on 1700 synthetic ST segments with baseline wander and 50 μV_{p-p} residual powerline interference: (a) Mean RMSD error vs cut-off frequency (b) Standard Deviation RMSD error vs cut-off frequency

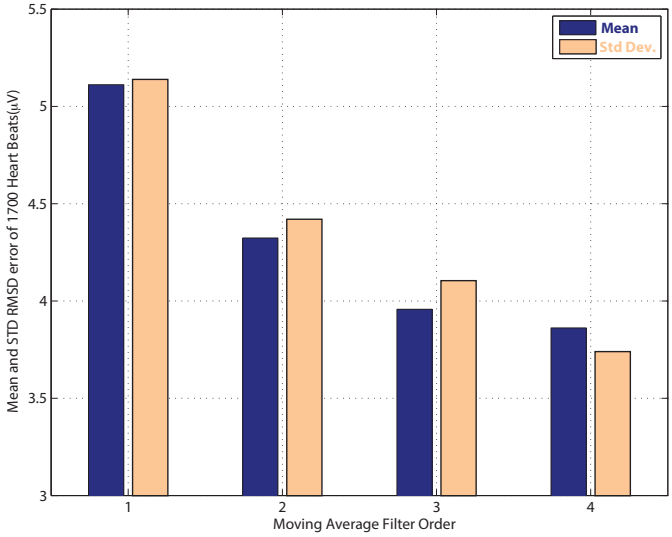


Figure 3.9.: Error vs order of moving average filter across 1700 synthetic ST segments added to filtered MIT-BIH Noise Stress Database baseline wander (BWM1.mat). Shown are mean and standard deviation of RMSD error

3.5.2.3. Moving Average Filter (MAF)

Moving averages, S_i, are the arithmetic means of n subsequent terms defined in a given sequence of data points, $(x_i, .., x_{i+n-1})$. Eq. 3.8 defines this relationship in mathematical terms. Reducing the number of data points with no aliasing affects the filter order which is inversely proportional with the downsampling rate, M.

Due to their nature, moving average filters are poor in separating one band of frequencies from another. However, they are suitable for time domain encoded signals and produce optimum results in reducing random noise while retaining sharp step responses [157]. Due to their smoothing ability, they are utilised for specific purposes in the ECG baseline wander detection algorithm.

$$S_i = \frac{1}{n} * \sum_{j=i}^{i+n-1} x_j \tag{3.8}$$

Suppressing the EMG interference to prevent multiple threshold crossings in ECG recordings can be achieved with a FIR MA filter design. In addition to removing these artefacts, other random noise sources such as the aliased noise due to downsampling and quantization noise is partially filtered and a smoother output is generated with MA filters.

The errors of different moving average filters are shown in Fig. 3.9. Here, it can be observed that a 3^{rd} order MA filter achieves similar results compared to higher order filters. However, a 4^{th} order filter implementation requires less computational complexity since the multiplication can be handled using logical shift operations. Thus, a 4^{th} order moving average filter is selected here.

3.5.3. QRS Detection

Although the main QRS detection concept originated from Pan & Tompkins algorithm [148–150], the parameters mentioned in Eq. 3.2, 3.3 and 3.4 are investigated thoroughly and they are adapted to the baseline wander detection algorithm.

3.5.3.1. Differentiator

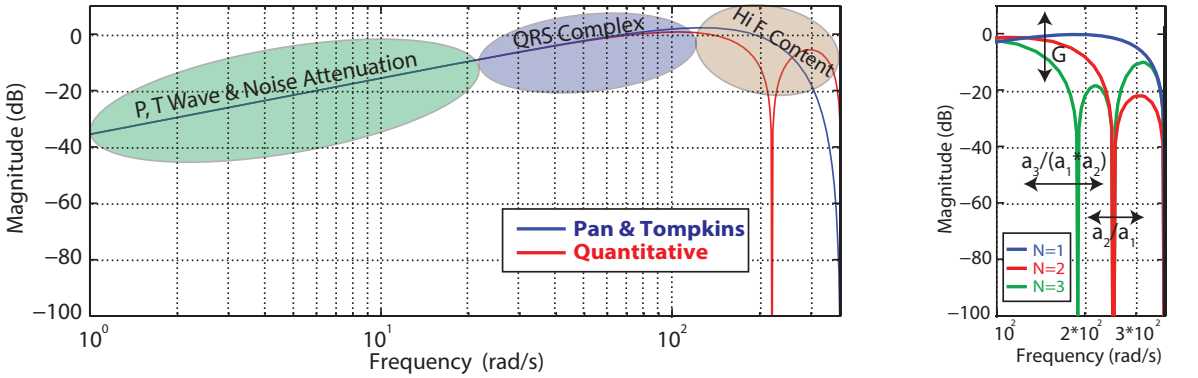
The main purpose of differentiation in QRS detection is to account for the steep slopes of the QRS complex such that R peaks can be detected in time through energy calculations and thresholding techniques. However, as ideal differentiators have a frequency response as in Eq. 3.9, their magnitude responses have a straight line enhancing higher frequency components. In discrete data systems, ideal differentiators require infinite bandwidth and even so their ideal behaviour is not desirable for most practical signals mainly due to SNR degradation [158]. Similarly, as in ECG signals most of QRS complex is situated within 5-40 Hz band and amplifying the noise at higher frequencies can drown out the desired signal.

$$H(e^{j\omega}) = j\omega \tag{3.9}$$

As the impulse response of such an ideal system is real and odd, an FIR approximation with the same form (Eq. 3.1) can be utilised as a differentiator. Substituting $z = e^{j\omega}$ in this equation yields an N^{th} order approximation with a gain and coefficients denoted as G and a_n respectively.

$$H(e^{j\omega}) = G * j \sum_{n=1}^{N} a_n * sin(n\omega) \tag{3.10}$$

This equation can be approximated to Eq. 3.9 and solved for coefficients to obtain an approximated version of an ideal differentiator. This includes solving for different parameters such as minima, mean squared error or maximal tangency mathematically. However, here the aim is to generate computationally efficient parameters while achieving a desired frequency response. In Pan & Tompkins algorithm, a_1 and a_2 is defined as 2 and 1 respectively [148]. In their quan-



(a) Pan & Tompkins differentiator magnitude responses

(b) Magnitude responses

Figure 3.10.: Differentiator Bode plots at f_s=200 Hz

titative analysis, however, these coefficients are defined the other way around [149]. In both articles the sampling frequency is defined as 200 Hz, whereas in baseline detection algorithm FIR filter differentiator approximations at a sampling frequency of 360 Hz are investigated.

Fig. 3.10 shows both approaches as reported in Pan & Tompkins publication and quantifies side lobe cut-off frequency parameters of a differentiator. Any differentiator defined utilising Eq. 3.10 suppresses the low frequency components (P and T waves) with respect to the QRS complex like an ideal differentiator. In addition, as a benefit of a non-ideal differentiator high frequency content does not get enhanced further and due to the high SNR ratio at high frequencies, non-ideal differentiators preserve signal quality. As these approaches are FIR implementations, their phase response is linear and the delay is defined by the order, N.

In the baseline estimation algorithm, unlike Pan & Tompkins, not all magnitude responses are suitable for implementation. This is due to the fact that downsampling reduces the sampling rate to $\frac{f_s}{M}$, causing some differentiator implementations to partially suppress the QRS complex. For instance, for a downsampling rate of 4, the quantitative magnitude response's side lobe appears to be around 25 Hz (N=2) which degrades system performance due to inaccurate QRS detection.

On the other hand, increasing the order of the differentiators increases the computational load, suppresses the QRS complex partially, and introduces larger delay. This can be seen with N=3 (7-point differentiators) as in Fig. 3.10(b). The ratio of the coefficients defines the side lobe frequency and for a downsampling rate of 3 such implementations do not improve the system performance. For this reason, a 5-point differentiator is implemented with coefficients a_1 and a_2 defined as 1 and 2 respectively (see Eq. 3.11). These coefficients are computationally efficient and calculations can be done with shifting operations and no distortion to QRS complex is observed with a downsampling rate of 3. Detailed analysis of N-point differentiators tested in system design and their magnitude response Bode plots are provided in Appendix B.1. For higher downsampling rates, 3-point differentiator implementations preserve QRS complex and thus avoid errors in energy collector calculations of the QRS complex.

$$y(nT) \approx \frac{2 * x(nT) + x(nT - T) - x(nT - 3T) - 2 * x(nT - 4T)}{8} \tag{3.11}$$

3.5.3.2. Moving Window Integrator

Following the differentiator, the signal is squared and passed through a moving window integrator. Governing energy operations are mentioned in Eq. 3.3. Here, the window size of the moving integrator is decided based on nominal ECG characteristics, downsampling rate and sampling frequency. The maximum window size for QRS complex lasts less than 120 ms [28]. Therefore, based on a 360 Hz sampling frequency and a downsampling rate of 3, a window size of 15 samples ($15 \geq 120$ ms X 120 Hz) is required to achieve a saturated output as shown in Fig. 3.11.

As the previous stages suppress the noise, attenuate P and T waves, and the squaring op-

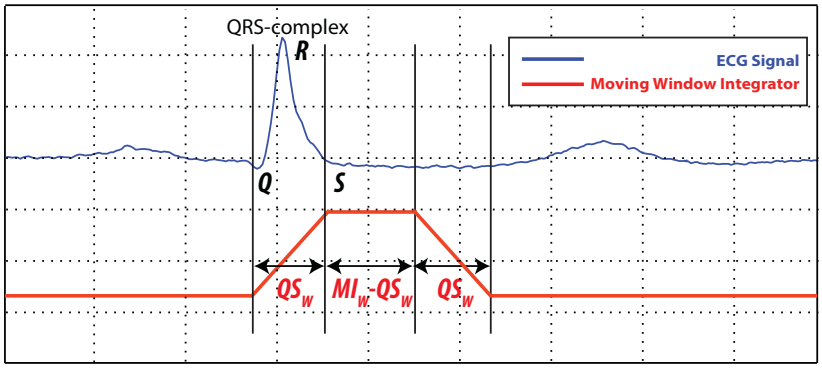


Figure 3.11.: Moving window integrator response. Notations denoted as: QS_W refers to QRS complex width and MI_W refers to integrator window size

eration increases the effect of steep slopes of the QRS complex, the moving integrator window size is defined based on the half of the maximum QRS complex. Once the algorithm detects a threshold crossing on the integrator output, then a QRS flag is generated; therefore preserving an ideal integrator response has no added benefit. This way the generated output is at least equal to the ideal integrator response as the output is normalised by the window size. On the other hand, the overall computational resources utilised in integration are reduced by eliminating extra additions and normalizing can be achieved with shifting operators.

In experimental testing, a moving window integrator with a window size of 15 generated the same number of QRS flags when compared to a window size of 8. Therefore, the latter has been implemented in the baseline wander detection algorithm.

3.5.3.3. Threshold Generation

Threshold coefficients are determined according to Eq. 3.4 in Section 3.3.3. Due to the adaptive nature of the threshold generation equation, an exhaustive search has been applied to the overall algorithm to determine optimal coefficient values, a and b. Fig 3.12 shows the error on ST segments of 1700 heart beats on the 16^{th} order MA filtered BWM1.mat data. As can be seen in Fig. 3.12, higher coefficient values, a and b, degrade the system performance. This is due to raising the new threshold higher than the upcoming peak subsequently resulting in missing fiducial point detections. The results match our previous work [156] where the best results are obtained with $a = 0.425$ and $b = 0.075$. To implement these as shifting operations, however, coefficient values of $a = 0.5$ and $b = 0.125$ are more computationally efficient choices and results using these coefficients do not significantly impact the errors.

One can think that P and T waves can affect threshold generation. However, nominal characteristics of these waves before pre-processing show that T wave amplitude is approximately equal to the half the size of the QRS complex whereas P wave amplitude is typically below 0.25 mV with longer durations [28]. These differences, when combined with the differentiator magnitude response (attenuating these waves approximately by 20 dB) and the squaring operation (further enhancing QRS to T & P wave SNR ratio), prevent triggering threshold detection

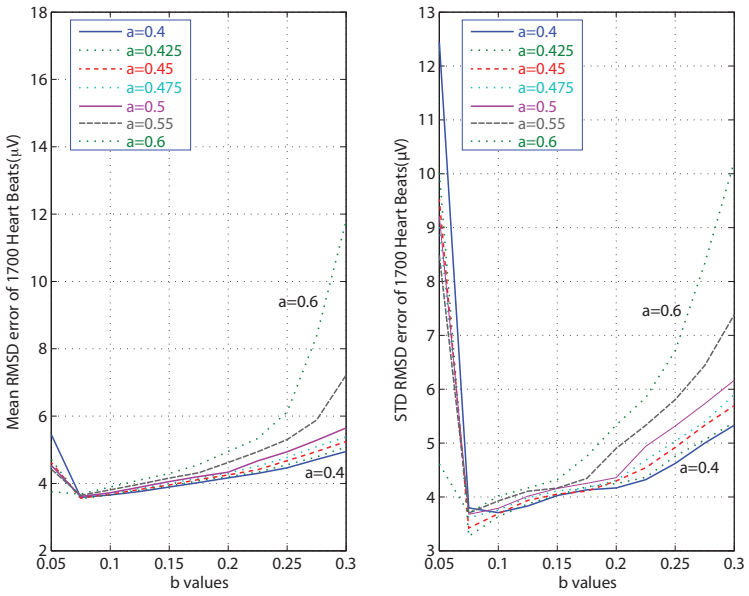


Figure 3.12.: RMSD error vs coefficient values across 1700 synthetic ST segments using 16^{th} order MA filtered MIT-BIH Noise Stress Database baseline wander (BWM1.mat). Shown are mean (left) and standard deviation (right) of RMSD error

of P and T waves.

3.5.4. Fiducial Point Detection

This section discusses the design parameter selection in detecting and locating fiducial points. As QRS detection is acquired by the previous stage, this information is utilised here to locate isoelectric points, *J1, J2* and *J3*. This process is illustrated in Fig. 3.13. The algorithm can be divided into three sections that are colour coded: (1) Watchdog Operation (red); (2) P and T wave detection (brown); and (3) Fiducial point search (green).

3.5.4.1. Recovery Operation

Here, a recovery operation has been implemented as the initial block of the fiducial point detection stage. The main purpose of this section is to recover and re-initialise the algorithm when no heart beat is being detected. Such conditions can occur in case of extreme motion artefacts that are large in magnitude and have high slopes. These interferences cannot be filtered thoroughly by the high-pass filter stage and the residual interference triggers a false positive QRS detection. In some cases, the new threshold might be set so high that the new upcoming peaks cannot trigger threshold crossings (see Eq. 3.4). During these type of stall events, the algorithm is recovered as illustrated in Fig. 3.13. If no QRS complex is detected for 1.5 seconds which corresponds to 40 bpm, the algorithm re-initialises the threshold. For a downsampling rate of 3, and a sampling frequency of 360 Hz, a counter continuously checks QRS detections and in the event of no detection for 180 samples, the threshold is set back to 0.

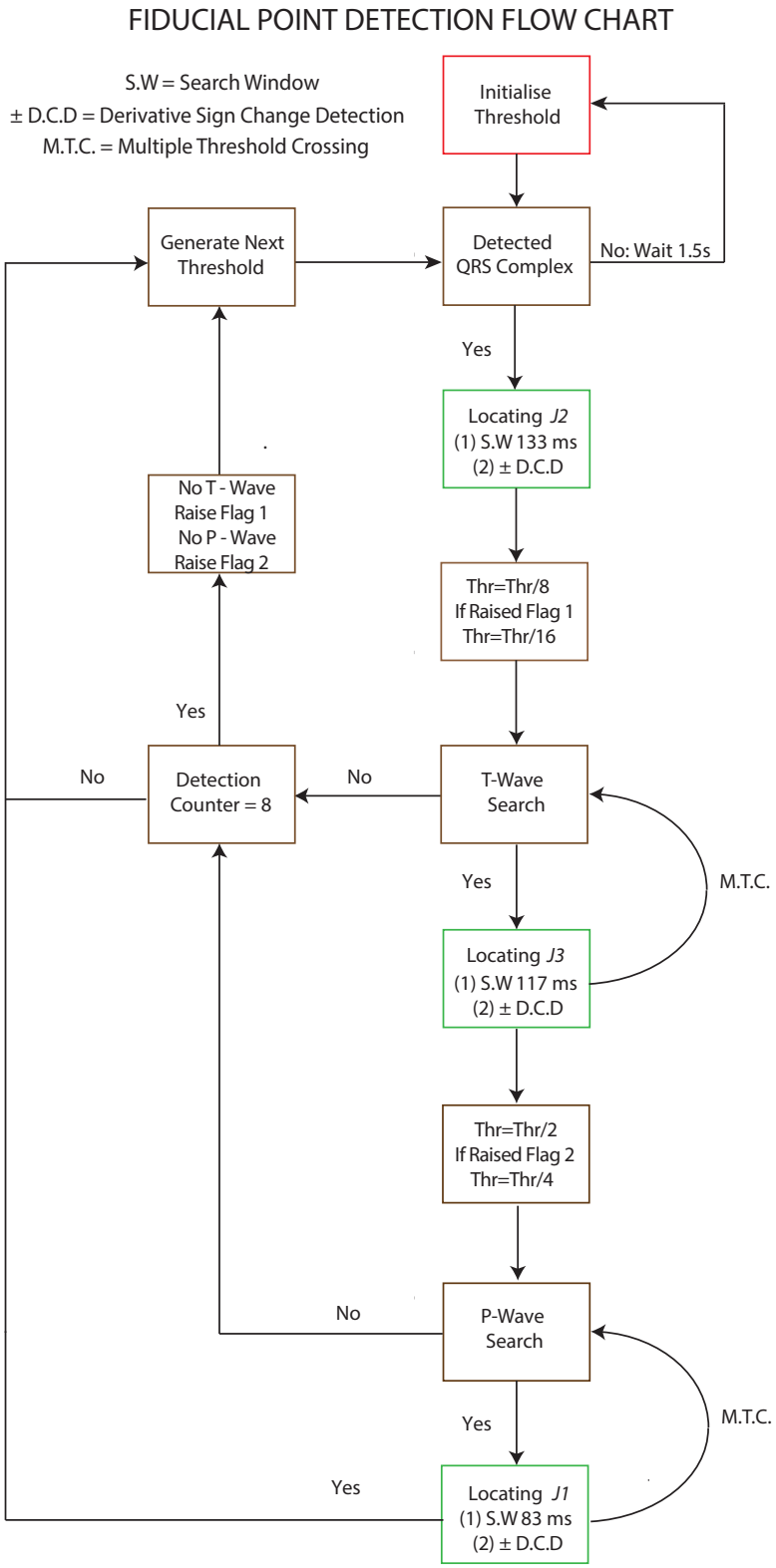


Figure 3.13.: Flow chart illustrating the fiducial point detection algorithm

3.5.4.2. P- and T- Wave Detection

As mentioned, the algorithm is divided into three stages and their purpose focuses on initialisation/recovery, detection, and search respectively. In this section, the detection methodology is discussed.

The detection stage locates QRS, P and T waves and this is achieved by generating the threshold as mentioned in Eq. 3.4. Once a wave detection generates a flag, the algorithm proceeds into fiducial point search. This will be discussed in detail in the following section. For simplicity and ease of understanding, however, it may be assumed that once a wave (QRS complex, P or T wave) is detected, the corresponding fiducial point, *J1, J2* and *J3*, is then also located. Such an example can be seen in Fig. 3.2, indicating a detected R peak, and the ST segment fiducial point *J2* is estimated when the signal derivative changes sign after a 60–80 ms delay after the S point.

The main aim of P and T wave detection stage is to locate these waves. This can be achieved by adjusting the threshold that has been generated for the QRS detection. As the T wave amplitude is usually equal to the half the size of QRS complex and P wave amplitude is less than 0.25 mV [28], using fractions of 1 / 8 and 1 / 16 of the QRS threshold to detect P and T waves respectively generates the best results, which are suitable for the tested MIT-BIH Arrhythmia Database signals.

In certain conditions, i.e. when P and T waves are substantially small, the algorithm has to adjust the detection further by a fraction of 1 / 2 when the algorithm misses to detect P and/or T waves. These instances are covered in the flowchart as shown in Fig. 3.13. When the algorithm does not detect T or P waves for 8 consecutive cycles, Flag 1 or Flag 2 is raised respectively. Once a raised flag is recognised, the new threshold fractions, 1 / 16 and 1 / 32, are utilised to detect these waves respectively. This way, the amplitude variances in P and T wave of ECG signals are taken into consideration during algorithm design.

3.5.4.3. Fiducial Point Search Window

Before proceeding further, the reader should know that the remainder of the algorithm utilises the time stamps of P-, T- waves and QRS complex detections that are employed to locate fiducial point search on the input sampled at 360 Hz.

Once the P, R and T waves are detected, corresponding fiducial points are searched as in this section. In the flowchart, these blocks are green coded and as can be seen in Fig 3.13 each search window utilises 2 criteria: (1) The duration of the search window; and (2) Derivative sign change requirement.

Search windows are defined as follows:

- *J2* search window is initiated after the QRS complex and lasts for 48 samples corresponding to 133 ms in duration. This search window is composed of two stages; (1) Wait Period; and (2) Locating Period. Once a QRS detection occurs, the algorithm waits for 18 samples to initiate the *J2* isoelectric point search. This wait period is determined by two

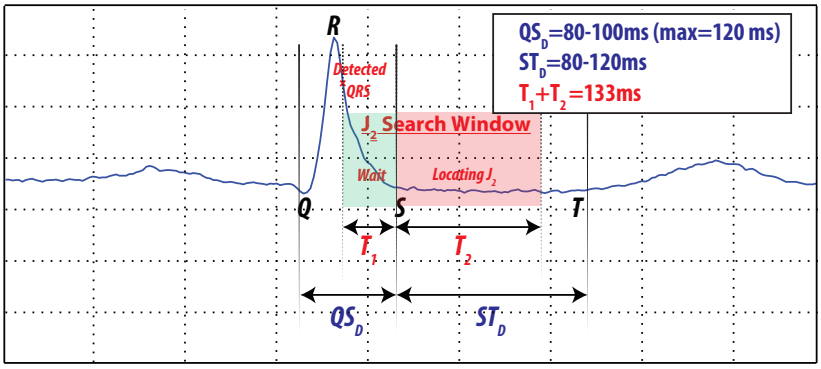


Figure 3.14.: *J2* search window definition

factors: (1) The maximum duration of the QRS complex defined ($\approx$120 ms), and (2) QRS detection location (after the R peak, close to the peak). Therefore, 50 ms of wait period is more than enough to end the QRS complex. On the other hand, the duration of the *J2* point window is defined by the duration of the ST segment. As this segment ranges from 80 ms to 120 ms, a window of 30 samples approximately lasting for 83 ms is defined to locate the fiducial point. The search window process for *J2* fiducial point is illustrated in Fig. 3.14. As can be seen, the total time of the search window corresponds to 133 ms ($T1 + T2$).

- Similarly, the *J3* search window starts after the T wave detection. As the nominal T wave duration is 160 ms, and the T wave is detected after the peak, 24 samples corresponding to 67 ms have been chosen as the wait period. Following this period, the search is initiated similar to the QRS search window, and *J3* is located within 18 samples (corresponding to 50 ms). Here, the search window is shorter than the *J2* search window. This is mainly because the RR interval defines the rest period between T offset and P onset and for increased heart rates the duration of these sections substantially decreases. While working with MIT-BIH Database signals, *J3* fiducial points are easily detected within these limits. The total time used for the *J3* search window corresponds to 117 ms ($T1 + T2$) as noted in the flow chart.

- Finally, the *J1* search window is initiated after the detected P wave. As the P wave lasts less than 120 ms, the search window starts 33 ms (12 samples) after the detection. To determine the duration of this search window, PR segment ranges (50 - 120 ms) are utilised and the search window is set for 18 samples, corresponding to 50 ms. The total time used for *J2* search window corresponds to 83 ms ($T1+T2$) as noted in the flow chart.

During search window operations, the second criterion locates *J1, J2* and *J3* fiducial points. This is achieved by detecting sign changes of the derivative within the defined search window.

3.5.4.4. Group Delay

Once the fiducial points are located, (depending on the filter types used), the estimated baseline wander is delayed by the overall group delay. The overall group delay required by the system is defined by the moving average filter, the differentiator and the integrator window size. A fourth order MA filter with a downsampling rate of 3 requires the system output to be delayed by 16 ms. Similarly, the five-point differentiator is an FIR filter approximation introducing a delay of 2 samples, which corresponds to an additional 16 ms. The delay of the moving integrator on the other hand corresponds to 8 samples which is equivalent to 67 ms, and the overall delay adds up to 100 ms.

In real time implementation, these group delays are required to be taken into consideration as well as the delay introduced by the total number of operations at the clock speed. However, one benefit of the system is: Once fiducial point search windows are defined, there is a 50 ms wait period in all three cases. Therefore, the overall system is required to be delayed by 50 ms (corresponds to 18 samples at 360 Hz) as defined by the Eq. 3.12.

$$\tau_{Total} = \tau_{MA\,Filter} + \tau_{Differentiator} + \tau_{Integrator} + \tau_{Search\,Window} \tag{3.12}$$

3.5.5. Baseline Wander Estimation

The final stage of the baseline wander estimation utilises the information gathered from previous stages, and estimates the baseline wander through interpolation. Below, each sub stage is explained in more detail.

3.5.5.1. Fiducial Point Shift

Up to this point, it has been assumed that fiducial points *J1, J2* and *J3* are isoelectric; however, in reality this is not the case, especially for patients with a previous history of heart attack as shown in Fig 3.15. In such cases, the heart tissue gets damaged and the J2 point becomes elevated or depressed. Therefore, assuming that these fiducial points are at the same elevation and estimating the baseline in such a manner can lead to errors. In the literature, single fiducial point cubic spline approaches exist and they utilise the PR interval (*J1*) as generating the interpolation points [88]. These locations do not alter in magnitude as much and have less effect on the ST segment while interpolating; however the ST segments' magnitude carries information and changes can be observed in this segment due to certain conditions. Therefore, during recording and processing elaborate design is required to detect these discrepancies.

To avoid introducing error in baseline estimation based on fiducial point discrepancies, the algorithm utilises elevation/depression differences of each interval maintained on the clean ECG signal at the start up (for each heart beat between *J1 & J2 and J2 & J3*). These relative magnitude differences of 8 averaged consecutive sections are stored temporarily in a variable and fed back to the input signal before interpolation takes place on the input. Due to the presence of noise interferences on the raw data such as white noise and baseline wander, discrepancy

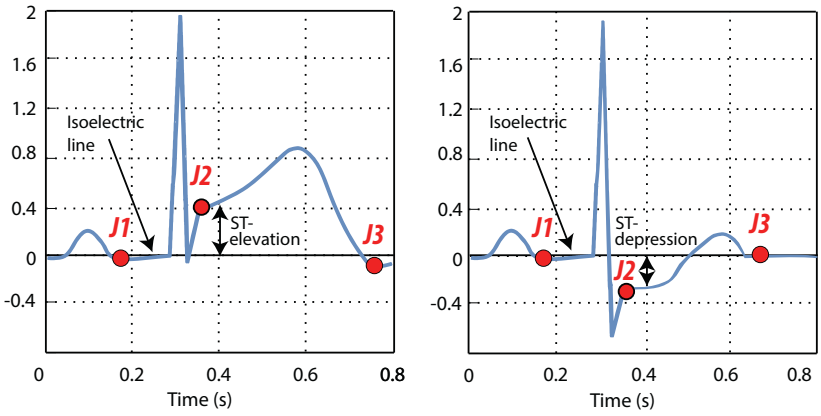


Figure 3.15.: *J1, J2* and *J3* fiducial point discrepancies

calculations are evaluated on the clean ECG signal. These interferences contaminate the signal of interest and usually discrepancy information resides below the noise floor. Filtered data are also not suitable for such an operation since the phase distortion introduced by IIR filters distorts these segments and minor elevation changes can not be captured as presented in Fig. 3.16.

As mentioned, 8 clean consecutive heart cycles are utilised to generate the segmental discrepancies present within the signal. These segmental differences are evaluated at the fiducial point locations after each wave detection. For instance, once a P wave is detected, the algorithm waits for the corresponding fiducial point to determine the PR interval level and utilises this information with the ST segment level detected after a QRS detection. Their difference generates the PR interval and ST segment discrepancy, which is denoted as *J1 & J2* difference. Similarly, once a T wave is detected, corresponding isoelectric level is located after a successful fiducial point detection and utilised with the ST elevation level to generate *J2 & J3* difference in the same manner. These generated differences are then compensated and baseline estimation is achieved on the shifted fiducial points.

In the event of unexpected large amplitude changes in the ST segment, a control mechanism is required. If the patient experiences a shift above 100 µV for 8 consecutive cycles at the ST segment as shown in Fig 3.15, the algorithm requires re-initialisation of discrepancy calculations. Even though filtered data are not suitable to capture minor changes, they are effective to detect large shifts. This approach is demonstrated in Fig. 3.16 with two different heart signals (with and without ST depression) showing their filtered responses as well as the raw data with baseline wander. Here, the patient on the right has ST depression whereas the patient on the left sub plot has a normal recording. When their filtered plots are compared, the segmental change can be detected and these changes are then compared with the temporary variable requiring re-initialisation to avoid removing elevated/depressed sections.

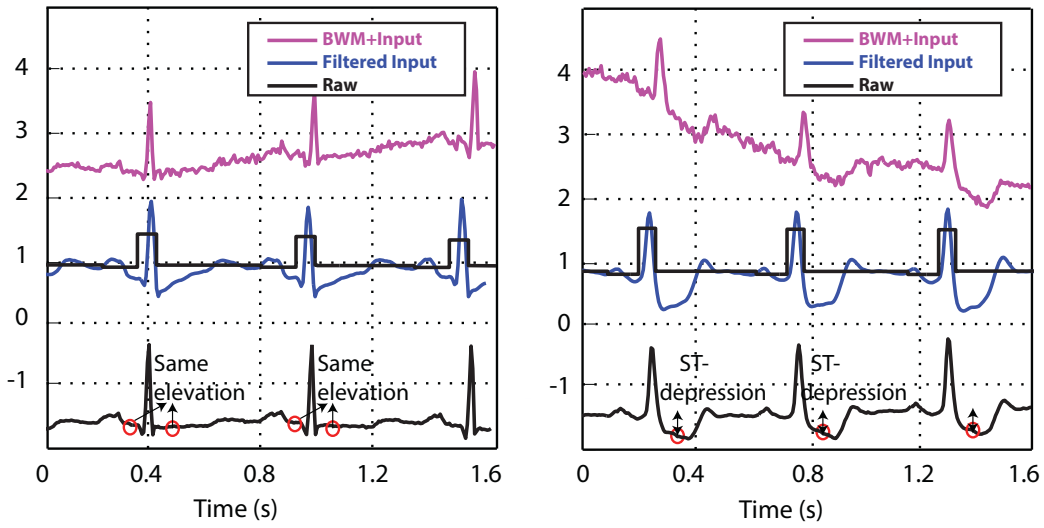


Figure 3.16.: Effect of ST elevation/depression changes on fiducial point discrepancy calculations. Shown are: (1) Patient (Left) with no elevation/depression on ST segment; (2) Patient (Right) with ST segment depression

3.5.5.2. Interpolation Methods

Once fiducial points *J1, J2* and *J3* are detected and elevation differences are compensated, the next step estimates the baseline wander through interpolation. Here, three different methods are compared namely cubic spline, piecewise cubic hermite, and linear interpolation and one is implemented as the interpolation method.

Tests have been performed on the synthetic data with added baseline wander signal, BWM1, acquired from MIT-BIH Noise Stress Database as mentioned in the evaluation metrics. Therefore, the ground truth is known and the histogram plots of each interpolation method are illustrated as in Figs. 3.17 and 3.18. These plots show RMSD error per ST segment and per heart beat over 1700 heart beats, whereas the highlighted areas in Fig. 3.18 demonstrate the large errors specific to each interpolation method.

As can be seen from the mean and standard deviation results, the overall error in each interpolation method is comparable with the other approaches. When their computational complexity is considered, linear interpolation requires less hardware resources as compared to its polynomial counterparts. Also, it should be noted that the high standard deviation errors in each interpolation method are due to present quantisation noise and EMG interference in the raw data. As the baseline wander estimation method does not suppress these at the output, the overall noise floor is defined by these type of noise interferences. Additionally, five instances of the recording show large step changes in the range of 4-5 mV. Even though these sudden shifts are eventually compensated by the algorithm, errors of corresponding heart beats are not included in these histogram plots to accurately compare each method. Based on the accuracy results, real data tests have been carried out with cubic spline interpolation.

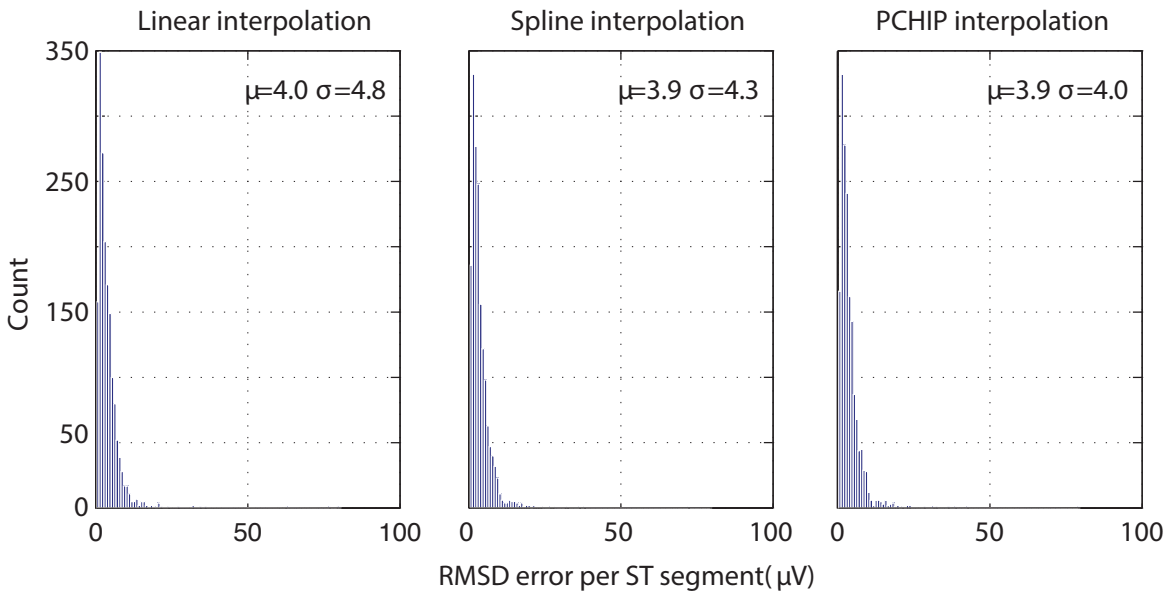


Figure 3.17.: Root mean square (RMS) errors per heartbeat for 1700 ST segments results with MIT-BIH Noise Stress Database baseline wander added (BWM1.mat). Results shown for 3 different interpolation methods sampled at 360 Hz.

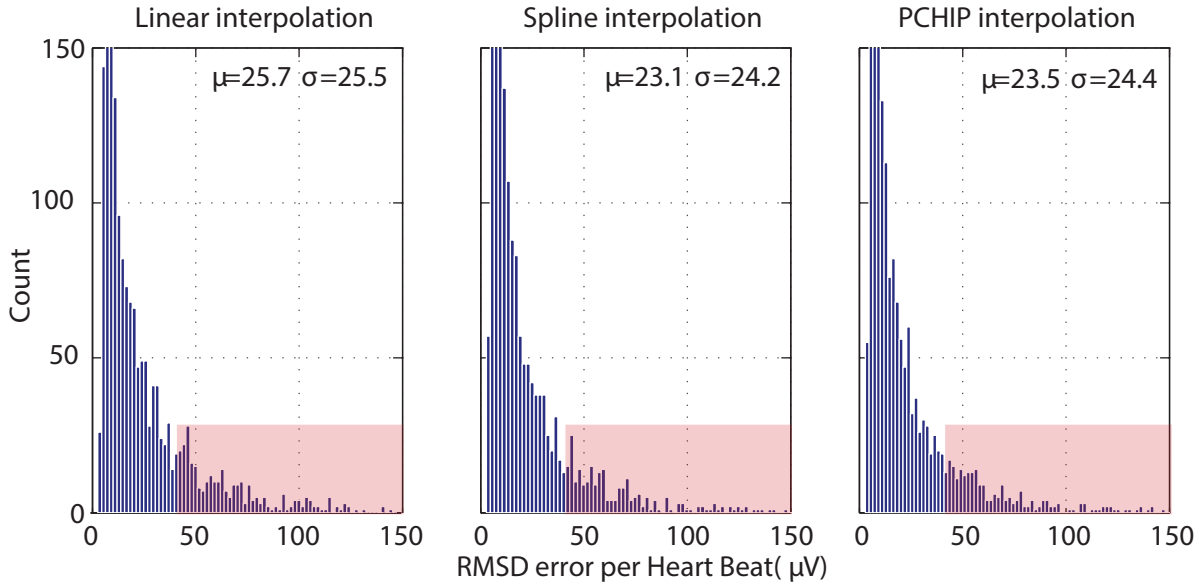


Figure 3.18.: Root mean square (RMS) errors per heartbeat for 1700 synthetic heart beats results with MIT-BIH Noise Stress Database baseline wander added (BWM1.mat). Results shown for 3 different interpolation methods sampled at 360 Hz.

3.6. Results & Discussion

In the preceding section design parameters are set and implementation of the baseline detection algorithm has been discussed in detail. Based on these set parameters, extensive tests have been carried out as will be described in this section. These tests involve synthetic and real data acquired from MIT-BIH databases as explained in Section 3.4.2. After a description of the tests, obtained results will be presented and discussed with comprehensive analysis comparing the algorithm performance with other methods. Finally, the results section is concluded with computational complexity analysis investigating approximate hardware resource requirements for the baseline wander detection algorithm.

3.6.1. Synthetic Data Analysis

As indicated in Section 3.4.2, synthetic data analysis has been carried on the ECG recordings generated from Fourier series approximation. Here, these recordings are combined with real baseline wander signals acquired from the MIT-BIH Noise Stress Database. Therefore, these test are only partially realistic. The overall aim is to carry out segmental analysis precisely on a synthetic ECG signal to evaluate the algorithm accuracy on the real baseline wander as the ground truth is known throughout the process.

3.6.1.1. Synthetic Data Test Results

The real baseline wander signals, namely *BWM1* and *BWM2*, are combined with the generated synthetic ECG signals. Table 3.1 shows the error in estimating baseline wander of the proposed algorithm at various SNR levels. These SNR ratios are recorded at ambulatory settings and can be added to any ECG recording at different attenuation levels in amplitude (at 0,6,...,24 dB). The resulting, RMSD errors are generated and tabulated within the S-T segments of the synthetic ECG data which is accurately time stamped as mentioned in Section 3.4.3.

Following these tests, other MATLAB filter implementations are compared with baseline wander detection algorithm. The filtering approaches in these tests involve low-pass elliptic IIR filters (utilising embedded *filter* and *filtfilt* functions in MATLAB), equiripple FIR implementation and the baseline wander estimation algorithm covered in this chapter. The aim is to compare baseline wander estimation algorithm with non-/linear filtering. As elliptic filters provide sharp cut-off and narrow transition width, they are chosen as non-linear phase response filters and also zero phase filtering with this filter is utilised with *filtfilt* MATLAB function. This function enables to filter the signal in both forward and backward directions [159]. Even though these systems are not causal, their generated off-line results are compared with the baseline wander detection algorithm. These filters estimate a 16^{th} order moving average filtered version of the BWM1.mat signal and their corresponding RMSD results are shown in Table 3.2. Additionally, correlation coefficient results of the estimated (denoted as B_e) and the real baseline wander (denoted as B_r) of each method are calculated as in Eq.3.13, where E denotes the expected value, μ denotes the mean and σ denotes the standard deviation.

$$\rho_{(B_e,B_r)} = \frac{E[(B_e - \mu_{B_e})(B_r - \mu_{B_r})]}{\sigma_{B_e}\sigma_{B_r}} \tag{3.13}$$

Table 3.1.: Synthetic data with real baseline wander RMSD errors

Dataset (Synthetic/ Real)	Attenuation (dB)	RMSD error (µV) (Without motion artefact)			Total beats	Interval (Err)
		μ	Med	σ	#	ϵ
BWM1 (S)	0	17.6	14.6	12.7	1681	S-T
BWM1 (S)	6	8.9	6.3	7.6	1681	S-T
BWM1 (S)	12	4.5	3.3	3.8	1681	S-T
BWM1 (S)	18	2.3	1.7	1.9	1681	S-T
BWM1 (S)	24	1.3	1.0	1.0	1681	S-T
BWM2 (S)	0	12.4	11.5	5.2	1681	S-T
BWM2 (S)	6	6.2	5.8	2.6	1681	S-T
BWM2 (S)	12	3.2	2.9	1.3	1681	S-T
BWM2 (S)	18	1.7	1.5	0.7	1681	S-T
BWM2 (S)	24	1.0	0.9	0.5	1681	S-T
Average	-	**5.9**	**5.0**	**3.7**	**1681**	S-T

Table 3.2.: Comparison table of different filters with baseline wander algorithm

Filtering Method	Filter Order	Synthetic Dataset	RMSD error (µV) (Without motion art.)			Correlation coefficient	Total beats	Int. Err
			μ	Med	σ	$\rho_{(B_e,B_r)}$	#	ϵ
LPF Elliptic IIR (filter)	7	MA Filtered BWM1	10.3	8.0	12.9	0.9840	1681	S-T
LPF Elliptic IIR (filtfilt)	7	MA Filtered BWM1	6.8	5.5	5.6	0.9933	1681	S-T
LPF FIR Equirriple	12218	MA Filtered BWM1	6.5	5.3	5.1	0.9940	1681	S-T
This work	-	**MA Filtered BWM1**	**3.8**	**3.4**	**3.6**	**0.9946**	**1681**	**S-T**

3.6.1.2. Synthetic Data Test Discussion

Synthetic data tests in Table 3.1 show that baseline wander estimation accuracy is directly proportional to the noise interference levels. As the amplitude ratio of the baseline wander is halved (every additional 6 dB attenuation), the mean/median/standard deviation RMSD errors at the ST segment approximately halve as well. This also explains the reason why BWM1 and BWM2 RMSD results differ in magnitude. A statistical analysis shows that BWM1 signal varies more in amplitude and conveys higher kurtosis results.

Synthetic test results reveal that upon successful fiducial point detection, the algorithm detects baseline wander accurately and even though these noise artefacts are sometimes 500% of the ECG signal peak to peak, baseline wander estimation is still within the limits as required by AHA and IEC [17–20]. It should be noted, however, that these results do not include isoelectric point elevation differences, white Gaussian noise in the ECG recording, and EMG noise artefacts. In cases where the noise floor is defined by these interferences, fewer fiducial points per heart beat are detected and this degrades the system performance. Therefore, real data tests are required to understand the effects of other noise sources on the overall algorithm accuracy.

Table 3.2 shows the results of the FIR and IIR filter implementations (with and without zero phase filtering) evaluated in MATLAB with double precision filter coefficients. These results are then compared with the baseline wander detection algorithm with double precision, whereas single precision implementation of the overall embedded system is covered in Chapter 5. It can be seen that the high order equiripple FIR filter (as described in Section 3.4.3) yields similar results compared to the 7^{th} order elliptic IIR filter with zero phase filtering. However, zero phase filtering is non-causal and requires data storage. On the other hand, low-pass elliptic IIR filter implemented with *filter* function in MATLAB, shows the least accuracy due to its non-linear phase response. Also it should be noted that, as synthetic ECGs have ideal ST segment responses with no low frequency content close to the transition bandwidth, distortion to this segment is limited. In real recordings, however, such disturbances are more pronounced as can be seen in Fig. 3.16, denoted with the blue ECG signal.

Finally, the baseline wander detection algorithm generates the most accurate results compared to other methods. The correlation coefficient is the closest to unity among all approaches showing high resemblance with the real baseline wander signal. Also it should be noted that, as the detection is done on the input, electrode offset is removed accurately.

3.6.2. Real Data Analysis

Following synthetic test, real data analysis has been carried on the MIT-BIH Arrhythmia Database (section 3.4.2). These signals typically contain step changes, EMG and motion artefacts, with variation in the isoelectric lines, missing P waves and irregular heart beat characteristics. Each of these events creates a challenge to any algorithm and, since these noise sources are not addressed specifically, a degradation in accuracy is expected when compared to synthetic test results.

3.6.2.1. Real Data Test Results

Table 3.3 and 3.4 show RMSD errors of the baseline wander estimation algorithm utilising MIT-BIH Database signals. The former presents results for the signals recorded in an ambulatory setting and annotated by at least two cardiologists. The latter presents the MIT-BIH Noise Stress Database baseline wander added to MIT-BIH Arrhythmia Database signal with varying signal to noise ratios, as described previously.

In these tests, the ground truth is not known and for that reason the error is calculated against a high order FIR filter (section 3.4.3) since this is the gold standard, (i.e. it is clinically accepted to use such filters). Average results show that when motion artefacts are not present, isoelectric fiducial point estimation provides clinically valid data with average mean, median and standard deviation RMSD errors of 28.7 µV, 25.8 µV and 15.4 µV respectively.

As more baseline wander is introduced into the system, BWM1.mat file, the algorithm performance degrades as shown in Table 3.4. These results are in agreement with the synthetic data analysis as covered in the preceding section, and as the attenuation level of the noise interference increases, its effect on baseline wander estimation diminishes. Dataset number 100 shows mean, median and standard deviation RMSD errors of 32.3 µV, 28.1 µV and 17.9 µV with introduced BWM1 signal (0 dB attenuated). Similar results can be observed with the dataset number 101, however, at higher attenuation levels the effect of the added noise artefact diminishes and mean errors are determined by the random noise present in the recordings.

3.6.2.2. Real Data Test Discussion

Real data tests have been carried out on 12 MIT-BIH Arrhythmia Database signals as reported in Table 3.3 and 3.4. Some of these recordings show a high number of motion artefacts compared to others therefore results with and without these noise interferences vary from signal to signal. The effect of these noise artefacts can be analysed from the mean and median results of each recording. Database signal 101 shows RMSD error change of 0.2 µV in its median with and without motion artefacts whereas the mean changes by 7.4 µV. When these results are interpreted in conjunction with the database signal 105, the difference in each result is much more significant and is related to the high number of motion artefacts present in the recording.

High standard deviation results with motion artefacts are explained by different reasons. In database signal 102, the high standard deviation results as well as the high mean and median are related to the precordial (chest) recording. This signal is recorded with the V5 chest lead and even though this lead recording generates the same structure as the Lead-II recording in a 12-Lead ECG system, P waves are substantially small when compared to the Lead-II counterpart. Therefore, when the algorithm does not detect P waves, the baseline estimation is acquired with a missing fiducial point. Consequently, the mean, median and standard deviation results degrade in all cases. On the other hand, database signal 105 shows large mean and standard deviation due to motion artefacts and also it should be noted that the signal is corrupted partially and no ECG signal is recognisable at these instances which degrades overall system performance. Finally, database signal 108 shows the worst standard deviation among all recordings. The

Table 3.3.: RMSD errors of MIT-BIH Database signals

Dataset (Synthetic/ Real)	Amp. Att. (dB)	RMSD error (μV) (With motion art.) μ	Med	σ	(Without motion art.) μ	Med	σ	Total beats #	Int. Err ϵ
100 (R)	0	14.8	13.0	11.0	14.6	13.0	8.2	2243	P-T
101 (R)	0	31.0	21.6	104	23.6	21.4	12.6	1835	P-T
102 (R)	0	46.5	35.1	41.2	37.0	32.3	21.0	2147	P-T
103 (R)	0	20.4	16.9	21.0	18.5	16.8	10.9	2044	P-T
105 (R)	0	89.2	31.8	168	32.7	27.1	19.6	2542	P-T
108 (R)	0	115	35.5	480	34.1	28.1	22.2	1733	P-T
111 (R)	0	43.1	37.4	29.2	39.9	36.6	19.9	2094	P-T
112 (R)	0	30.9	30.3	14.7	30.8	30.2	14.3	2509	P-T
115 (R)	0	29.7	24.4	20.9	28.1	24.1	16.6	1923	P-T
121 (R)	0	57.5	45.4	47.9	44.9	42.8	20.1	1833	P-T
122 (R)	0	18.8	17.5	9.5	18.6	17.5	8.8	2446	P-T
123 (R)	0	23.0	20.2	24.0	21.3	20.2	10.4	1488	P-T
Average	-	**43.3**	**27.4**	**81**	**28.7**	**25.8**	**15.4**	**2070**	P-T

Table 3.4.: RMSD errors of MIT-BIH Database signals with added baseline wander

Dataset (Synthetic/ Real)	Amp. Att. (dB)	RMSD error (μV) (With motion art.) μ	Med	σ	(Without motion art.) μ	Med	σ	Total beats #	Int. Err ϵ
100+BWM1 (R)	0	37.1	28.6	53.6	32.3	28.1	17.9	2243	P-T
100+BWM1 (R)	6	24.2	19.6	33.3	22.4	19.6	13.4	2243	P-T
100+BWM1 (R)	12	18.9	16.1	16.1	18.2	16.1	10.7	2243	P-T
100+BWM1 (R)	18	16.7	14.8	11.8	16.4	14.8	9.4	2243	P-T
100+BWM1 (R)	24	16.0	14.0	11.6	15.8	14.0	9.1	2243	P-T
101+BWM1 (R)	0	44.8	30.4	106	33.8	29.5	18.7	1835	P-T
101+BWM1 (R)	6	32.8	21.4	101	24.6	21.0	14.8	1835	P-T
101+BWM1 (R)	12	28.2	17.5	102	20.5	17.2	13.2	1835	P-T
101+BWM1 (R)	18	26.7	16.0	103	19.2	15.8	13.0	1835	P-T
101+BWM1 (R)	24	26.1	15.6	103	18.7	15.5	12.6	1835	P-T
Average	-	**27.2**	**19.4**	**64.1**	**22.2**	**19.2**	**13.3**	**2039**	P-T

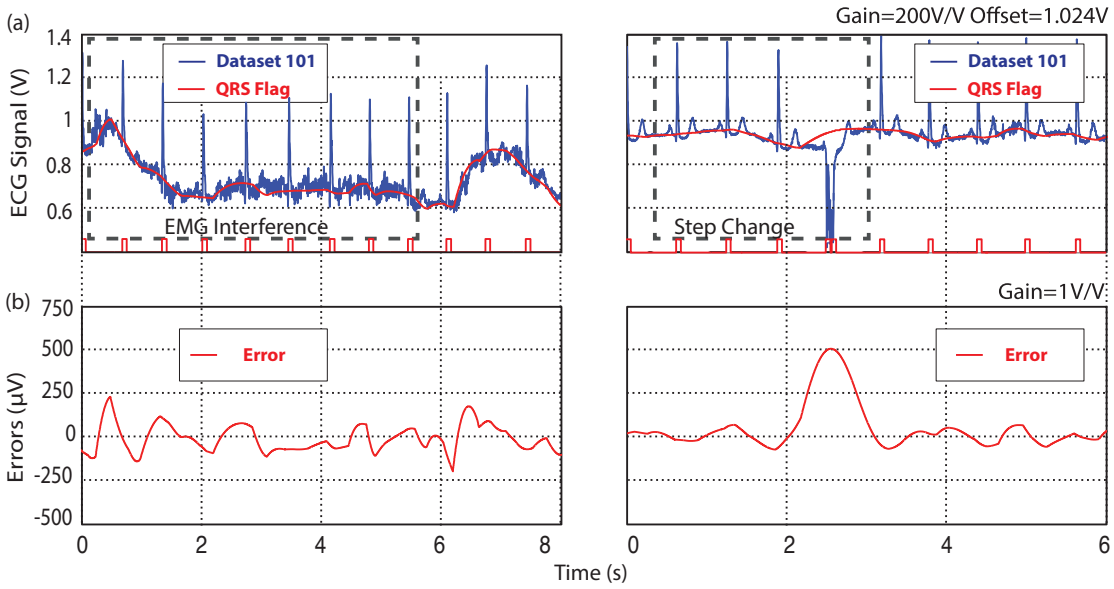


Figure 3.19.: MIT-BIH Arrhythmia Database signal - 101.mat (a) distortion variations when EMG and step changes are present. QRS flags are also shown. (b) RMSD error (difference between estimated vs high order FIR filter)

channel exhibits considerable noise and baseline shifts as well as inverted QRS complexes which means the septum depolarizes from right to left meaning the right ventricle contraction is triggered first and followed by the left ventricle. These QRS complexes do not trigger threshold crossings, and therefore, at these instances the algorithm fails to track baseline wander.

On the other hand, some signals have quantisation noise present with small P waves. These make detection of all three fiducial points a challenge as mentioned before. One such signal is database signal 121 such that mean and median RMSD results of these signals are determined by the quantization noise floor which substantially suppresses small P waves (approximately around 50 µV) at various instances.

During large EMG interference, the noise floor increases substantially. At these instances, accurate detection of P and T waves becomes more challenging and the algorithm performance degrades due to fewer fiducial point detections. On the other hand, when a step change occurs, the algorithm detects it as a QRS complex and initiates the *J2* point search. However, if a real QRS complex is detected before locating the fiducial point, the algorithm proceeds with no detection, thus can not correct narrow step changes. The error related to the EMG interference and narrow step changes is shown in Fig 3.19. Similar to step change errors, the algorithm requires a successful fiducial point detection during motion artefacts and any delay in successful detection results in accuracy degradation [156]. These events increase the mean, median and standard deviation RMSD errors to 43.3 µV, 27.4 µV and 81 µV respectively.

When comparing results with and without motion artefacts, it can be seen that median errors are similar, which implies that the large standard deviation is due to the motion artefacts and step changes. These errors are present in any filtering approach as the motion artefacts contain

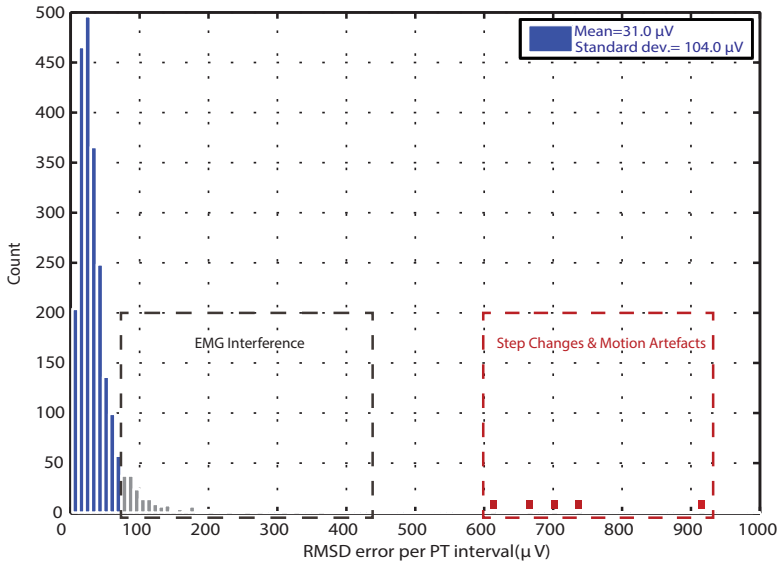


Figure 3.20.: RMSD error per P-T interval histogram plot of MIT-BIH Arrhythmia Database signal - 101.mat

frequency components that overlap with the ECG signal (as shown in Fig. 3.1). Fig. 3.20 shows an error histogram for the entire MIT-BIH Arrhythmia 101.mat dataset with 1835 P-T intervals. In this data set, approximately 100 heartbeats are contaminated with EMG interference and 5 heartbeats undergo step changes and motion artefacts. These errors are annotated on Fig. 3.20 with different colour markers. It should be noted that these errors are denoted at P-T intervals since the clinically annotated files only cover this information. When these interferences are suppressed, the results clearly show that RMSD errors are well within the AHA and IEC standards of a 100 µV variation in the ST segment.

The work of Afsar *et al.* [160] and additional analysis of Fasano *et al.* [96] lists accuracy results of baseline wander estimation of various algorithms on data signals acquired from European Society of Cardiology (ESC) ST-T Database as in Table 2.3. Similarly, Fasano and Villani list MIT-BIH Arrhythmia Database results on ST segments and they are presented in Table 2.4.

3.7. Conclusion

As discussed, clinical baseline wander detection algorithms are computationally demanding and not suitable for real-time implementation. Most of the existing approaches rely on frequency-based methods and due to strict requirements by guidelines, they demand extensive computational resources.

This chapter has proposed a novel real-time fiducial-point based tracking approach to estimate the baseline drift that can be implemented on low-power hardware. This approach relies on time stamping 3 "isoelectric" fiducial points and estimating the baseline wander through interpolation. The design parameters are determined to reduce the computational complexity of the overall system with the aim of finding a good balance between resource efficiency and

accuracy.

It has been shown that the method can be used to remove the baseline drift without causing significant distortion in the ST segment and as such can be applied to achieve clinically-viable ECG waveforms. Through extensive tests on synthetic and real data, average RMSD errors with 5.9 µV mean, 5.0 µV median and 3.7 µV standard deviation, and 22.2 µV mean, 19.2 µV median and 13.3 µV standard deviation are measured respectively. Both of the synthetic (synthetic data with added baseline wander acquired from MIT-BIH Noise Stress Database) and the real (MIT-BIH Arrhythmia Database with added baseline wander acquired from MIT-BIH Noise Stress Database) test results reveal that baseline removal is within the guidelines stated by the AHA and IEC for clinically valid ECG and these tests do not significantly distort the sensitive ST segment.

In the event of large noise artefacts, the algorithm performance degrades due to the noise caused by these interferences even though fiducial point estimation in time is precise. Therefore, other methods suppressing these artefacts at the output without distorting ST segment integrity can improve the accuracy of the overall system.

Chapter 4

Piecewise Linear Interpolation For Non-Uniformly Sampled Biosignals

The previous chapter developed a new algorithm to estimate ECG baseline wander by locating the fiducial points in an ECG recording. These fiducial points are interpolated and baseline wander estimates are generated with different interpolation techniques to quantify the developed algorithm thoroughly.

There are of course several interpolation algorithms and methods reported in the literature and it has been an intensively studied subject, where each approach aims to approximate smoother curves and better fits. However, the extensive requirements of most of these algorithms limit their real-time applications and there lies a challenge to find a balance between their complexity, accuracy, and adaptability. The latter is crucial in biological applications as changing signal dynamics carry a challenge in system design. Therefore, one needs to understand the limitations of these interpolation algorithms and investigate their suitability for real-time biological applications as well as their computational complexity prior to hardware implementation.

It is the aim of this chapter to assess the suitability and/or limitations of different interpolation techniques and to propose a new method that is computationally efficient and suitable for interpolating non-uniform sampled biosignals. This approach focuses on improving the final stage of the baseline estimation algorithm covered in the previous chapter aiming to preserve available on-chip resources of the overall system. The chapter is organised as follows: Section 4.1 lists the main objectives, Section 4.2 provides a brief background on interpolation algorithms and describes the challenges in ECG baseline wander removal; Section 4.3 describes the overall system concept and methods; Section 4.4 covers the datasets and evaluation metrics briefly; Section 4.5 details the parameter selection and the implementation; Section 4.6 presents and discusses the results with complex algorithms; and Section 4.7 concludes the chapter.

4.1. Objectives

As mentioned previously, this chapter focuses on a new interpolation algorithm that is suitable for real-time signal processing applications. The requirements and the key objectives of this chapter can be summarised as follows:

- **Accuracy:** As the baseline wander can be modelled as a low frequency sinusoid signal with a varying fundamental frequency, the interpolation algorithm has to detect any curvatures and estimate the baseline wander without degrading the system performance.
- **Adaptability to Biological Signals:** Fiducial points are generated based on the ECG morphology such as detection of P, QRS and T waves. Therefore, these fiducial points are non-uniformly sampled and the interpolation algorithm has to interpolate through these points and take into account the variance of these fiducial points in time.
- **Suitability for Real Time Systems:** The main application aims for real-time systems; therefore, the algorithm has to estimate the baseline wander on the go without requiring the overall system to store large amounts of data.
- **Resource Utilisation:** Any operations requiring multiplication and/or division significantly contribute to power consumption. Therefore, a balance between complexity and accuracy needs to be identified such that the proposed algorithm can improve the overall system performance where needed.

4.2. Background

4.2.1. Evolution of Interpolation

Interpolation is the method of generating new data points within the range of a discrete dataset. In other words, it is the way of generating information that is not available explicitly within the signal itself. Historically, this way of retrieving information can be dated as far back as ancient Babylonian (300 BC) and Greek (190-120 BC) times [161]. During those times, linear interpolation was used to predict astronomical events and had impacts on certain practical needs such as farmers basing their strategies according to these estimations.

In the 17^{th} century, Newton initiated an advancement in Mathematics with his contributions. In his famous work, *Principia*, he published two formulae related to interpolation: one dealing with equal-interval data, and the other focusing on the more general case of arbitrary-interval data. As a continuation to Newton's general formula, Edward Waring mentioned one of these formulae without requiring the computation of finite and divided differences in his work [162]. Similarly, Lagrange who was unaware of Edward Waring's work at the time, published his representation 16 years later [163] and introduced the formulae we know of today as Lagrange polynomials. Following these years, more advancements have been noted with the age of scientific revolution. A detailed description of these advancements are covered in the work of Erik Meijering [161].

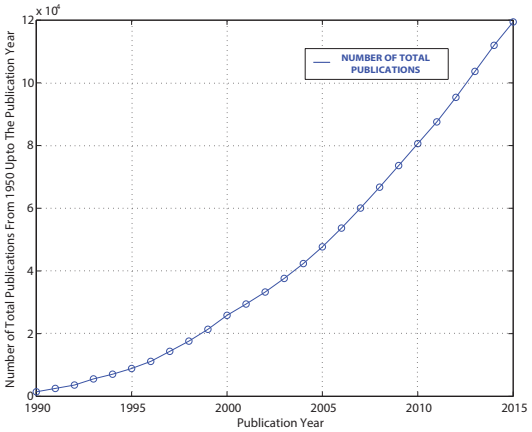


Figure 4.1.: Number of overall publications with the keyword "interpolation" in their title, list of keywords, or the abstract over the last two decades

Today, it is an area of interest for most studies as we have entered the era of digitization, where many applications require additional information whether it is required in processing, analysis or communication of information. This increased interest can also be observed by the overall number of publications containing the word "interpolation" in their title, list of keywords, or the abstract as revealed by the Institute for Scientific Information in the Web of Science as shown in the Fig. 4.1.

4.2.2. Interpolation Methods

As mentioned in the work of Erik Meijering [161], interpolation methods and techniques have been evolving continuously from the basic linear approach to polynomial and convolution based methods. As for our research, some of the basic methods that form the basis to our approach as well as other interpolation algorithms utilised to evaluate the proposed approach are covered in this section.

4.2.2.1. Polynomial Interpolations

Linear Interpolation The simplest of all the interpolation methods, linear interpolation, generates additional information by joining two data points through a straight line. This line follows the form of $y = mx + b$, where m is denoted as the slope of the straight line and b is the y-intercept. A more generalized formula is shown in Eq. 4.1, where (x_i, y_i) and (x_{i+1}, y_{i+1}) are the coordinates of two arbitrary data points and m is the slope between these two points. Fig. 4.2 shows the reconstruction of an arbitrary new data point at (x_N, y_N). This simplistic approach is appropriate only for "slowly varying functions" due to small variations in the signal.

$$y = m(x - x_i) + y_i, \ \ where \ \ \ m = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \tag{4.1}$$

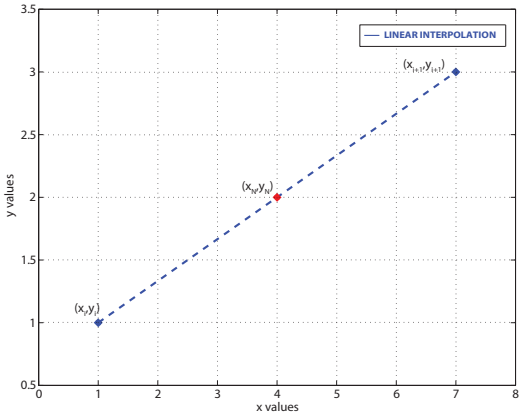


Figure 4.2.: Linear interpolation - Illustration of generating a new data point

Quadratic Interpolation Unlike linear interpolation, higher order polynomial interpolations are expected to yield more accurate approximations for "regularly varying functions". The next simplest interpolation method utilised for such signals is the quadratic interpolation and its generalized formula has the form as in Eq. 4.2, where the coefficients are denoted by a, b and c and two data points are denoted as (x_i, y_i) and (x_{i+1}, y_{i+1}).

$$y = a + b(x - x_i) + c(x - x_i)(x - x_{i+1}) \tag{4.2}$$

Lagrange Interpolation Polynomials Following quadratic interpolation, higher order interpolation polynomials are defined by Lagrange polynomial representation. This representation defines the least degree of a polynomial curve that passes through a given set of coordinates (x_i, y_i) as in Eq. 4.3. However, due to the nature of this representation, any small perturbations in coordinates result in large overshoots at the end points, known in the literature as the Runge Phenomenon [164]. These oscillations have no relation with the true nature of the defined function and due to this effect higher order polynomial interpolations both degrade accuracy and increase complexity of the interpolation method.

$$fx = f(x_0)\frac{(x - x_1)(x - x_2)(x - x_3)...}{(x_0 - x_1)(x_0 - x_2)(x_0 - x_3)...} + f(x_1)\frac{(x - x_0)(x - x_2)(x - x_3)...}{(x_1 - x_0)(x_1 - x_2)(x_1 - x_3)...} + ... \tag{4.3}$$

4.2.2.2. Spline Interpolations

The errors associated with higher order polynomials due to small perturbations in coordinates led to development of spline interpolation methods. These are piecewise defined polynomial functions that are connected at the interpolation coordinates also known as knots.

Cubic Spline Interpolation One such example is the cubic spline interpolation which has been well accepted by achieving a smooth representation of the signal and preserving the continu-

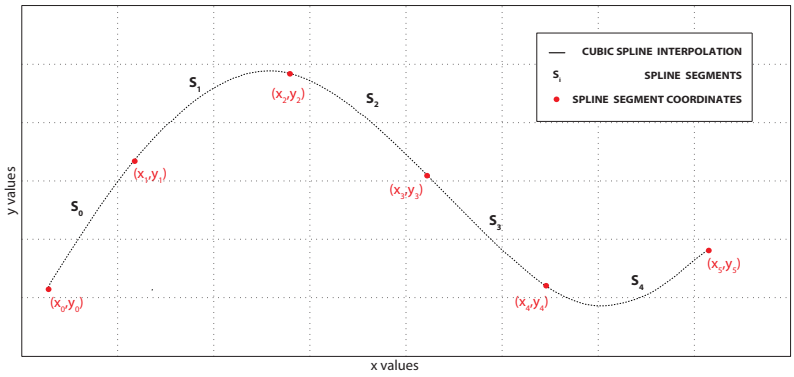


Figure 4.3.: Cubic spline interpolation for each knot defined

ity of the derivatives. Eq. 4.4 shows the general representation of the cubic spline interpolation, where the spline segment is denoted by $S_i(x)$ and the coefficients of each segment are denoted by a_i, b_i, c_i and d_i. Cubic spline representation with the predefined knot sequence, (x_0, y_0) to (x_5, y_5), is illustrated in Fig. 4.3.

$$S_i(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i \tag{4.4}$$

In order to achieve cubic spline interpolation, one has to make sure splines and their first & second derivatives are continuous at the knot locations. If there are $M + 1$ number of vertices defined, M number of spline segments require $4M$ equations to be solved such that each spline coefficients are determined, a_i b_i, c_i and d_i. One can obtain $4M - 4$ number of equations at $M - 1$ interior vertices by using the conditions defined in Eq. 4.5.

$$\begin{aligned}
&Condition\ (1) \rightarrow S_i(x_i) = y_i \\
&Condition\ (2) \rightarrow S_{i-1}(x_i) = y_i \\
&Condition\ (3) \rightarrow S'_i(x_i) = S'_{i-1}(x_i) \\
&Condition\ (4) \rightarrow S''_i(x_i) = S''_{i-1}(x_i)
\end{aligned} \tag{4.5}$$

Additionally, two more equations are defined at the end points, $S_0(x_0) = y_0$ and $S_{M-1}(x_M) = y_M$ and the remaining two equations can be defined by equating the second derivatives to zero at those end points to obtain a natural cubic spline approach. There are additional ways of maintaining a spline approach discussed in the literature, where the first or the third derivatives at the end points are fixed to zero rather than the second as in the works of Carl De Boor and George Elmer Forsythe respectively [165], [166]. When all conditions are utilised, $4M$ equations form a triangular matrix which can then be solved by "forward elimination" and "backward substitution" to evaluate spline parameters. A detailed explanation of such solution can be found in the works of Beatty and *et al.* [167].

Piecewise Cubic Hermite Interpolation Piecewise cubic hermite interpolation interpolates a given set of data points via certain subroutines. Even though cubic spline interpolation produces more accurate results, the key idea of PCHIP interpolation is to avoid overshooting by knowing both the function and first derivative values given a set of data points. These first derivatives, d_k, as mentioned in Moler's work are defined as follows [168]: Let k be an interior point within a set of data points, when δ_{k-1} and δ_k have opposite signs or equal to zero, d_k is equal to zero, whereas when they have the same sign, d_k is equal to the harmonic mean of those two discrete slopes as formulated in Eq. 4.6.

$$\frac{1}{d_k} = \frac{1}{2} * (\frac{1}{\delta_k} + \frac{1}{\delta_{k-1}}),\ where\ \ \delta_k = \frac{y_{k+1} - y_k}{x_{k+1} - x_k} \tag{4.6}$$

On the other hand, when the distance between $x_{k+1} - x_k$ to $x_k - x_{k-1}$ is not equal, then the relationship mentioned in Eq. 4.6 becomes a weighted harmonic mean of those distances. Due to the nature of this approach, continuity of the second derivatives, C^2, can not be guaranteed. However, PCHIP generates a monotonic output and might be preferred over cubic spline approach if the data consists of "steep" and "flat" sections. Therefore, their effect might be of interest when investigated in ECG signals while estimating the baseline wander.

There are of course other techniques such as B-spline, exponential spline interpolation and many more [165], [169]. However, due to their computational complexity these methods are not covered here in detail.

4.2.3. Non-Uniformly vs Uniformly Sampled Interpolation Techniques

Interpolation of non-uniformly sampled points is often a challenge in signal processing as most approaches rely on uniformly spaced data. However, techniques exist to overcome periodically missing samples of a sampled signal sequence as long as Nyquist-Shannon criterion is met.

One such example is by utilisation of filterbanks with fractional delay filters as building blocks. These utilise windowed sinc methods to approximate the fractional delays with FIR subfilters [170], [171]. These types of filter bank implementations are used for reconstruction of periodically non-uniformly sampled signals since such approaches avoid digital noise coupling to the analogue signal [172].

When missing interpolation points are not repeated sequences, or in other words, samples are not periodically missing but rather depend on the nature of the signal, generalized equation of B splines exists to recover the signal from its discrete sample set [173]. These spline functions have minimal support with respect to a given degree and it is possible to express any spline function with a linear combination of B-splines which requires irregularly spaced sample locations based on the integer multiples of the sampling period of the discrete signal.

Additionally, continuous time signals can be obtained from non-uniformly sampled data by utilising fluency sampling functions. These functions enable to correspond to time varying signals by changing classes. One of the benefits of such an approach is, unlike piecewise polynomial methods that require coefficient determining, fluency functions need only sample values to be convoluted where these pseudo samples are simply generated by linear approximation [174].

Moreover, iterative algorithms also focus on recovering band limited signals from their non-uniformly spaced samples [175–177]. However, these are computationally demanding algorithms with potential convergence issues.

4.2.4. Challenges

Baseline wander removal has crucial importance in ECG signal processing and challenges associated to baseline detection require preserving the signal integrity for ambulatory systems. These challenges encountered by any interpolation algorithm can be itemised as follows:

- The standards of American Heart Association (AHA) and International Electrotechnical Commission (IEC) allow distortions up to 100 µV at the ST segment [17–20]. Therefore, the interpolation algorithm has to estimate the baseline wander without introducing large distortions and must track the real baseline wander as accurately as possible.
- Due to bradycardia, single fiducial point estimations per heart beat show accuracy degradation as the distance between interpolation points increases [88]. A multiple fiducial point per heart beat approach aims to overcome this problem by shortening the distance between interpolation points. Such an approach, however, increases the computational load and requires compensation of isoelectric point differences. These additional measures need to be investigated to quantify baseline wander detection.
- Although smoothness of the baseline wander estimate can be achieved by higher order interpolation algorithms, the presence of high frequency content degrades the baseline wander estimation accuracy. Based on these noise artefacts, it is harder to detect fiducial points accurately and for that reason interpolation algorithms are required to find a good balance between computational complexity vs the accuracy of the system performance.
- Interpolation points are detected during different intervals based on various factors such as R-R interval, QRS complex and P & T wave durations of a patient; therefore, the interpolation algorithm has to interpolate non-uniformly sampled data and be adaptable to biological signals. These requirements limit the use of most conventional techniques in ECG baseline estimation.

4.3. Computationally Efficient WPL Interpolation Algorithm

ECG signals are prone to interference from physiological and environmental sources. In this section, we propose a new algorithm to estimate ECG baseline wander which is based on weighted piecewise linear (WPL) and linear interpolation. This approach utilises fiducial points detected by the baseline wander algorithm as covered in Chapter 3. As mentioned in Section 4.2.4, the proposed algorithm needs to comply with the clinical standards in ECG baseline wander estimation, reduce the computational complexity when compared to higher order spline and polynomial interpolation techniques, and be able to interpolate non-uniformly sampled data.

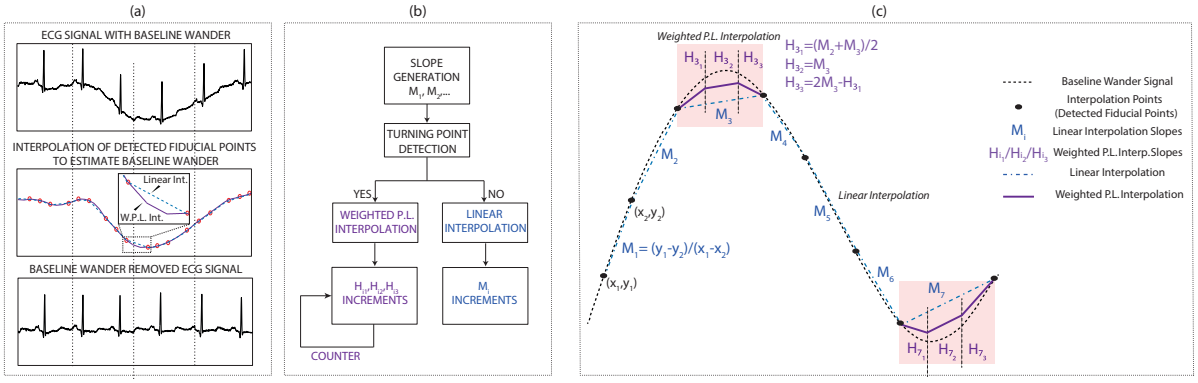


Figure 4.4.: Computationally efficient weighted piecewise linear interpolation methodology showing: (a) example input signal; (b) algorithm flowchart; and (c) illustration of concept

4.3.1. Methodology

The overall methodology of the computationally efficient WPL interpolation algorithm is illustrated in Fig. 4.4. The main purpose of the proposed algorithm is to minimise errors at the curvatures (turning points) with a better approximation than linear interpolation and simply in other cases, to focus on reducing the computational complexity of the overall system. Fig. 4.4(c) illustrates curvature approximation on a sinusoid at the peaks and valleys of the input signal as well as showing linear interpolation at other instances.

The interpolation algorithm is divided into two main stages: (1) Turning point detection and (2) Interpolation methods. The first stage detects possible curvature points on the input signal, whereas the second stage utilises this information to either focus on improving the accuracy or reducing the computational complexity of the overall system [178]. Fig. 4.4(b) shows the interpolation flowchart of the overall system.

4.3.2. Turning Point Detection

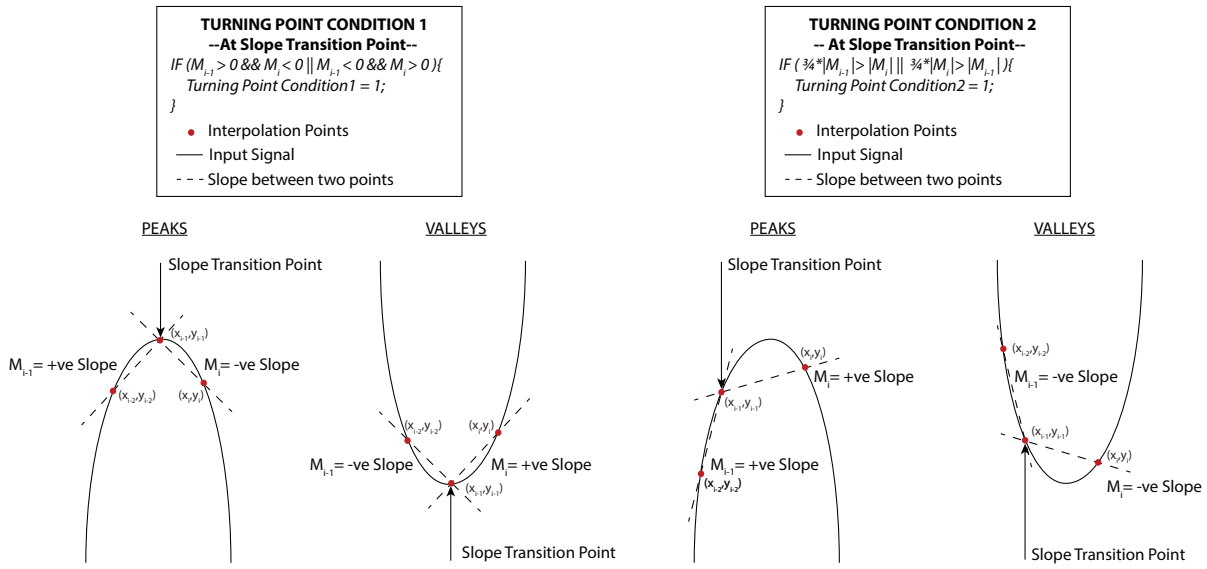
Initially, turning point detection stage generates the slopes, M_i and M_{i-1}, by using three adjacent interpolation points, (x_i, y_i), (x_{i-1}, y_{i-1}) and (x_{i-2}, y_{i-2}), as in Eq. 4.7. Then the algorithm checks for a turning point based on the predefined curvature detection conditions. Here we utilise two criteria:

$$M_i = \frac{y_i - y_{i-1}}{x_i - x_{i-1}}, \; M_{i-1} = \frac{y_{i-1} - y_{i-2}}{x_{i-1} - x_{i-2}} \tag{4.7}$$

4.3.2.1. Turning Point Detection Condition 1[1]

The first condition checks if there is a sign change between two consecutive slopes, M_i and M_{i-1}, as shown in Fig. 4.5(a). This sign change means an absolute/local maxima or a minima exists

[1]First detection rule based on the slope sign change detection



(a) Condition 1 (b) Condition 2

Figure 4.5.: Turning point detection conditions

in between the two coordinates, (x_i, y_i) and (x_{i-2}, y_{i-2}). Following such a detection, the output is then generated through WPL interpolation improving the overall accuracy of the system.

4.3.2.2. Turning Point Detection Condition 2[1]

On the other hand, sometimes the first condition on its own is not enough to detect the turning points sufficiently. This situation happens when (x_{i-1}, y_{i-1}) interpolation point is located before the peak and there is no detection of sign change in slopes. One example of such occurrence is shown in Fig. 4.5(b). In this example, both of the slopes, M_i and M_{i-1}, have the same sign and the curvature can not be detected based on the first condition only. Therefore, an additional condition checks for slope magnitude changes to avoid such cases and be still able to detect and interpolate these sections with WPL interpolation. Both of the conditions utilised throughout the interpolation algorithm are expressed in Eq. 4.8.

$$\begin{aligned} Condition\,1 &\rightarrow M_{i-1} > 0\,\&\,M_i < 0 \;||\; M_{i-1} < 0\,\&\,M_i > 0 \\ Condition\,2 &\rightarrow \frac{3}{4} * |M_{i-1}| > |M_i| \;||\; \frac{3}{4} * |M_i| > |M_{i-1}| \end{aligned} \tag{4.8}$$

4.3.3. Interpolation Methods

Based on the evaluation of these conditions, the algorithm either focuses on increasing the accuracy of the overall system in case of a turning point by applying WPL interpolation or

[1]Second detection rule based on the slope magnitude calculations

otherwise utilises linear interpolation to reduce the computational complexity of the system.

4.3.3.1. Linear Interpolation

As mentioned in Section 4.2.2.1, it is possible to write the general equation of a line for any two coordinates at any instant. However, this information is not required and increases the computational load while interpolating. A better approach is to add a fraction of the current slope, M_i, to every interpolated point in between y_{i-1} and y_i. This way the number of operations required is reduced as the slopes are already computed for turning point detection, and the interpolation is mainly based on addition operations. Fig. 4.4(c) shows the linear interpolation instances on an example input signal, $M_{1,2,4,5,6}$. One of the benefits of such an operation is being able to interpolate both uniformly and non-uniformly sampled data.

4.3.3.2. Weighted Piecewise Linear Interpolation

An improvement to linear interpolation is achieved when turning points are detected and curvatures are interpolated with WPL interpolation such as M_3 & M_7 instances as shown in Fig. 4.4(c). As can bee seen, these instances are divided into smaller segments, H_{i_1}, H_{i_2} and H_{i_3}, and interpolated with different linear functions to achieve better accuracy. To do so, two criteria need to be defined: (1) WPL Interpolation Equations and (2) Segmentation.

WPL Interpolation Equations When using linear interpolation, the continuity of the second derivatives, C^2, is not considered and peak/valley interpolation often carries a challenge. One solution to this problem is to divide the interpolation intervals into smaller segments and utilise better approximations with WPL interpolation equations when curvatures are detected. This technique originated from Euler's method referred to as Runge-Kutta method as discussed in the literature [179]. In this technique, Taylor series expansion of a function can be approximated as in Eq. 4.9 and it can be used to evaluate a function through its derivative function by taking a step size, h. As the step size gets smaller, then the estimated function values converge to the real function values. Fig. 4.6 shows the first order Runge-Kutta approximation with different step sizes on a sinusoid.

$$\begin{aligned} f(x_0+h) &= f(x_0) + f(x_0)' * h + \frac{f(x_0)'' * h^2}{2!} + ... + + \frac{f(x_0)^n * h^n}{n!} \\ &\approx f(x_0) + f(x_0)' * h \end{aligned} \tag{4.9}$$

Even though the full derivative function in WPL interpolation is not known as required in Runge-Kutta method, the slopes, M_{i-1} and M_i, are defined at the end points. With this information one can define slope functions, H_{i_1}, H_{i_2} and H_{i_3}. Here, the motivation is to generate a smoother transition than linear interpolation by reducing sudden changes in slope. This is achieved by utilising a weighted average of the known slopes, M_{i-1} and M_i, and then transitioning to the current slope, M_i, and finally compensating any overshooting/undershooting at

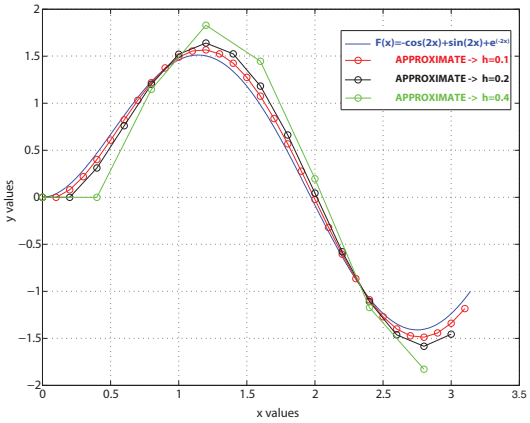


Figure 4.6.: First order Runge-Kutta method

the remaining section of the segment. Eq. 4.10 defines WPL interpolation equations utilised.

$$H_{i_1} = \frac{M_{i-1} + M_i}{2};\ H_{i_2} = M_i;\ H_{i_3} = 2 * M_i - H_{i_1} \tag{4.10}$$

Segmentation Based on the number of WPL interpolation equations defined as in Eq. 4.10, the segment, from x_{i-1} through x_i, is divided into smaller segments to apply each slope function to its matching segment. To do so, following the turning point detection, the distance between x_{i-1} and x_i is calculated and during WPL interpolation a counter continuously checks for duration of each segment, H_{i_1}, H_{i_2} and H_{i_3}, to avoid interpolation errors. In cases where equal segment partition can not be achieved, the algorithm introduces a compensation factor to the last sample such that interpolation point, (x_i, y_i), is always met. Even though higher segmentation is possible, this also requires additional segment equations, which increase the computational complexity of the algorithm with no or little added benefit.

WPL interpolation Once these two criteria are defined, corresponding slope segment increments, H_{i_1}, H_{i_2} and H_{i_3}, are added every clock cycle to generate WPL interpolation as shown in Fig. 4.7. The first segment's slope is the average of the M_{i-1} and M_i slopes. This enables a smoother transition rather than an instant shift between slopes whereas the second segment is interpolated with the original segment slope, M_i, and finally the last segment slope is introduced to meet the final interpolation points. Meanwhile, as mentioned before the counter checks the segment partition and makes sure that interpolation points are always met. As in linear interpolation, WPL interpolation can interpolate both uniformly and non-uniformly sampled data since each segment is interpolated with its defined slope functions, and is independent of previous segments. Due to the definition of these slope functions, the error function is bounded by the M_{i-1} and M_i. A detailed theoretical error analysis is carried in Section 4.6.

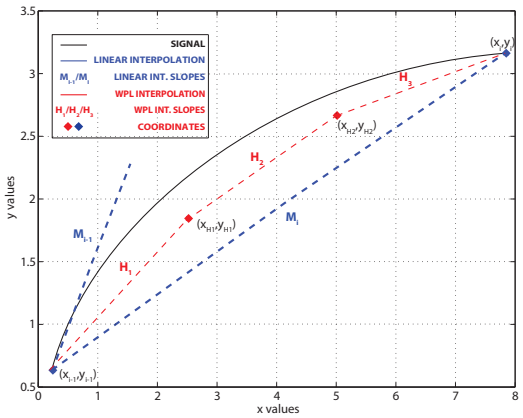


Figure 4.7.: WPL interpolation illustration with three segments.

4.4. Test Data & Evaluation Metrics

Two different software platforms, MAPLE and MATLAB are utilised for testing and validating WPL interpolation algorithm. MAPLE 2014 platform is used to verify the methodology such that the error function can be solved algebraically at any two arbitrary point in time and then validation tests have been carried in MATLAB R2015b platform both with synthetic and real data. A detailed explanation and characteristics of both synthetic and real data sets utilised for algorithm validation are covered in the previous chapter in Section 3.4.

Additionally, it should also be noted that the interpolation points that are used both in synthetic and real data WPL interpolation validation are generated by using MIT-BIH Arrhythmia Database signal, 100m.mat. These points are, therefore, realistic representations of non-uniformly sampled interpolation points.

Finally, both root mean square deviation (RMSD) errors and the maximum absolute error deviations of the estimated and the real baseline wander are evaluated as the accuracy metric, as covered in the previous chapter. The main reason of such an approach is the fact that RMS errors carry good measure of its effect for sinusoidal signals, whereas maximum error seen during ST segment carries crucial information as covered in the literature [17–20].

4.5. Design and Implementation

In this section, the sensitivity of key design parameters of WPL interpolation algorithm is quantified and investigated through parametric analysis. Following this investigation, results are then validated via MAPLE platform.

4.5.1. Turning Point Detection

As mentioned in the methodology section, two conditions continuously monitor turning points. When either one of these conditions is detected, WPL interpolation is utilised in baseline wander

estimation. Therefore, detailed analysis and quantification of these conditions are covered in this section.

Condition 1→ As mentioned, Eq. 4.8 shows that the first condition detects sign changes of two consecutive slopes. This sign change guarantees a local/absolute maxima or minima detection. The location of this maxima or minima on the other hand is not known precisely due to the lower sampling rate of interpolation points when compared to the input signal. Therefore, in cases where the preceding slope approaches to zero and the upcoming slope changes sign, turning point condition triggers WPL interpolation which might result in overshooting; such an example is shown in Fig. 4.8. This is due to few interpolation points being available per period and WPL algorithm lagging to detect curvatures accurately. To avoid these situations, an extra requirement in the overall condition function is added and tested on synthetic inputs as in Eq. 4.11, where the coefficient, a, checks the magnitudes of consecutive slopes. This way overshooting instances are aimed to be kept to a minimum by discarding such turning point detections and utilising linear interpolation at those instances instead.

$$Condition\,1 \rightarrow M_{i-1} > 0\,\&\,M_i < 0 \;||\; M_{i-1} < 0\,\&\,M_i > 0\,\&\,\mathbf{a} * |\mathbf{M_i}| > |\mathbf{M_{i-1}}| \tag{4.11}$$

Synthetic tests have shown that the number of these instances is related to the interpolation point sampling frequency and the input signal frequency. When a 0.1 Hz sinusoidal input is applied, 20 overshooting instances are observed over 2243 heartbeats and as the synthetic baseline wander frequency increased to 0.3 Hz and 0.7 Hz, the number of these occurrences increased to 50 and 180 respectively.

Tests showed that the accuracy improvement was limited and in some cases the added requirement condition performed even worse as shown in Fig. 4.9. It can be seen that, for low frequency inputs, WPL interpolation without the added requirement introduces less error than the anal-

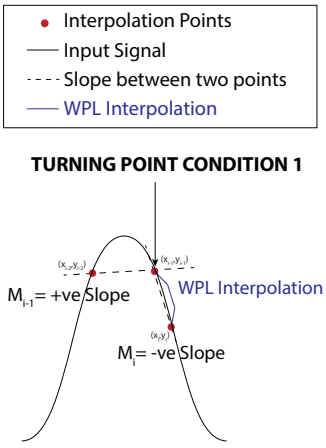


Figure 4.8.: WPL interpolation overshooting occurrence

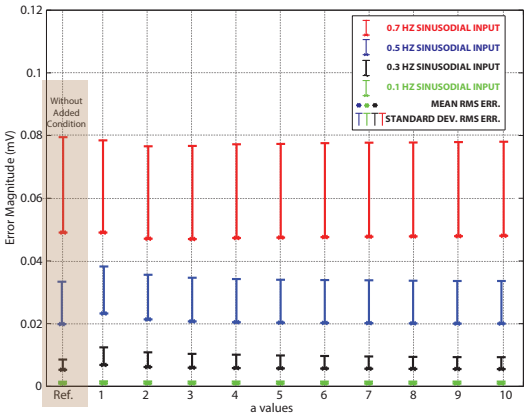


Figure 4.9.: Condition 1 with the coefficient, a. Parametric analysis with $1\,\mathrm{mV}_{p-p}$ synthetic sinusoidal inputs. Shown signals are 0.1 Hz (Green), 0.3 Hz (Black), 0.5 Hz (Blue), and 0.7 Hz sinusoidal input (Red) with mean and standard deviation RMS errors

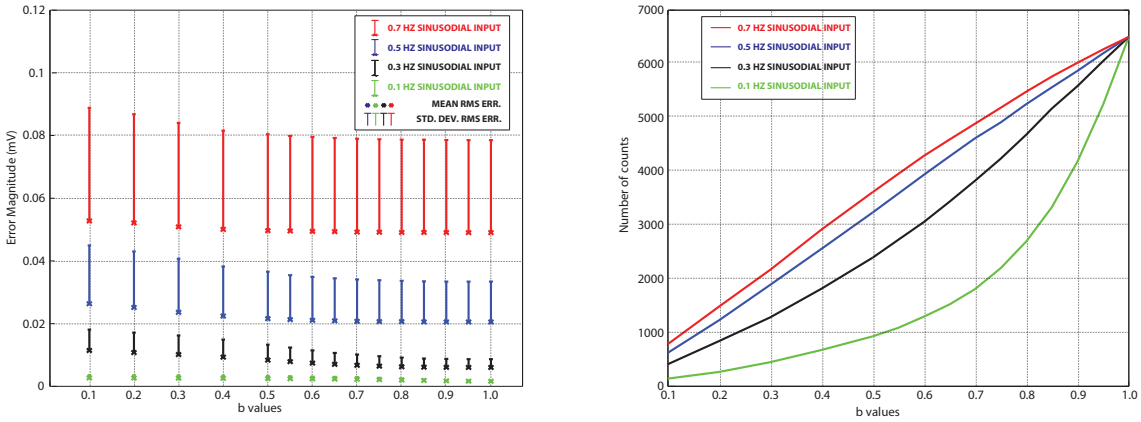
ysed version. In other words, the overshooting introduced by the WPL algorithm performs better than the undershooting linear interpolation estimate at those instances. However, as the frequency of the input signal increases and the interpolation point sampling frequency remains constant, accuracy improvement can be observed. This can be seen in Fig. 4.9 with the 0.7 Hz sinusoidal input. In cases, where undershooting has more importance and computational load is not a priority, this function can be implemented. However, as long as the interpolation point sampling frequency in relation to baseline wander frequency remains constant, the accuracy improvement is worse. Also it should be noted that, Condition 1 change also requires changes in Condition 2 introducing additional computational load as in Eq. 4.12. Therefore, the overall algorithm has been implemented without these changes due to limited accuracy improvement and increased computational load.

$$Condition\,2 \rightarrow \frac{3}{4} * |M_{i-1}| > |M_i| \;||\; \frac{3}{4} * |M_i| > |M_{i-1}| \;\&\; ... \tag{4.12}$$

$$(\mathbf{M_{i-1} < 0} \;\&\; \mathbf{M_i < 0} \;||\; \mathbf{M_{i-1} > 0} \;\&\; \mathbf{M_i > 0})$$

Condition 2→ As explained in Section 4.3.2.2, sometimes Condition 1 is not sufficient enough to detect curvatures; therefore, an additional condition is introduced based on slope magnitudes as in Eq. 4.8. Eq. 4.13 is the general form of this equation and the coefficient, b, in this equation, is determined based on parametric tests on synthetic data with various fundamental frequencies. Results of these tests are depicted in Fig. 4.10.

$$Condition\,2 \rightarrow \mathbf{b} * |M_{i-1}| > |M_i| \;||\; \mathbf{b} * |M_i| > |M_{i-1}| \tag{4.13}$$



(a) Accuracy vs. b values

(b) Complexity vs. b values

Figure 4.10.: Condition 2 - Parametric analysis with 1 mV_{p-p} synthetic sinusoidal inputs. Signals are denoted as 0.1 Hz (Green), 0.3 Hz (Black), 0.5 Hz (Blue), and 0.7 Hz sinusoidal input (Red)

Fig. 4.10(a) shows that the accuracy of the algorithm improves as b values increase with diminishing error magnitudes at higher b values for all tested sinusoids. For instance, 0.7 Hz and 0.3 Hz sinusoidal input RMS errors start to saturate around b values of 0.60 and 0.80 reaching its minimum at unity respectively. Even though the error difference between saturation point and minimum error point is marginal and computationally an unity approach seems reasonable, such a condition would trigger WPL interpolation always and computationally would be less efficient. Therefore, an event count analysis measures how many times WPL interpolation is triggered by each b values with different sinusoidal inputs over approximately 30 minute datasets. Fig. 4.10(b) shows the results of such analysis. The optimal b value requires interpreting accuracy and complexity plots together. In this study, b value is defined as 0.75 aiming to improve the system design by triggering WPL interpolation less when compared to unity b value and the multiplication defined in Eq. 4.13 is achieved by a shift and an addition operation.

4.5.2. Segmentation

In this section, the effect of segmentation between two interpolation points from x_{i-1} through x_i is investigated through parametric analysis without any structural change in WPL interpolation equations. Table 4.1 shows the number of partitions and tested function at each segment. It should also be noted that at higher segmentation orders, possible permutations of WPL interpolation equation arrangements are also taken into consideration. Fig. 4.11 shows the synthetic test results of each segmentation order utilising WPL interpolation equations as shown in Table 4.1. Here, the frequency of sinusoidal signals ($sin(\omega t)$) are varied from 0.1 to 0.7 Hz and the RMS mean and standard deviation errors are plotted against input frequency.

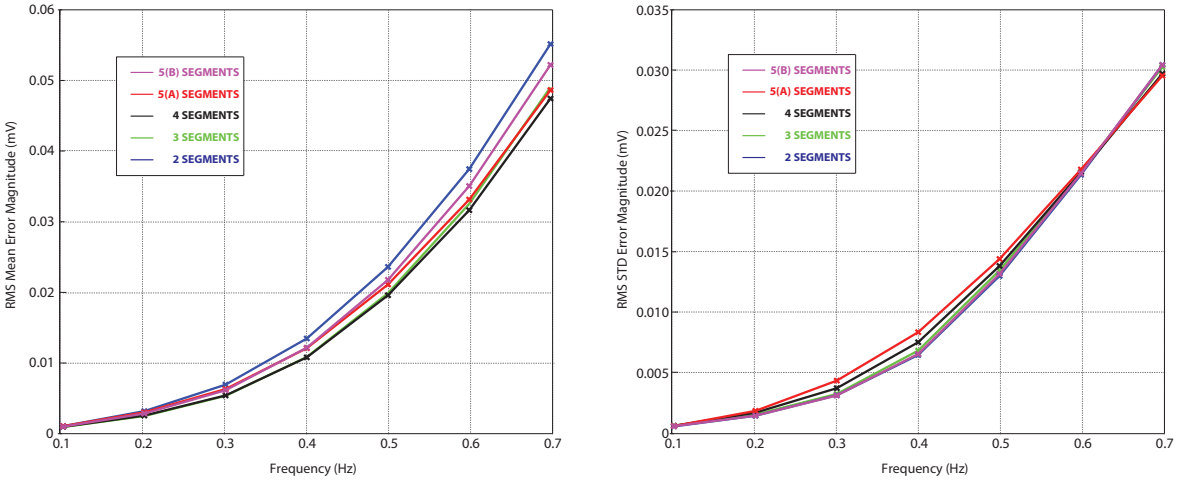
Fig 4.11(a) shows that the lowest mean RMS errors are achieved by 3^{th} and 4^{th} order segmentation. At lower frequencies, segmenting 3 times performs better; however, as the sinusoidal input frequency increases and interpolation point sampling frequency remains constant, 4^{th} or-

Table 4.1.: Segmentation and WPL interpolation equation of each segment

# Segments	First Segment Eq. (H_{i_1})	Second Segment Eq. (H_{i_2})	Third Segment Eq. (H_{i_3})	Fourth Segment Eq. (H_{i_4})	Fifth Segment Eq. (H_{i_5})
2	$\frac{M_{i-1}+M_i}{2}$	$2*M_i - H_{i_1}$	-	-	-
3	$\frac{M_{i-1}+M_i}{2}$	M_i	$2*M_i - H_{i_1}$	-	-
4	$\frac{M_{i-1}+M_i}{2}$	M_i	M_i	$2*M_i - H_{i_1}$	-
5 (a)	$\frac{M_{i-1}+M_i}{2}$	M_i	M_i	M_i	$2*M_i - H_{i_1}$
5 (b)	$\frac{M_{i-1}+M_i}{2}$	$\frac{M_{i-1}+M_i}{2}$	M_i	$2*M_i - H_{i_1}$	$2*M_i - H_{i_1}$

der segmentation becomes more accurate. This is due to the fact that as the distance between interpolation points, x_{i-1} and x_i, increases, the accuracy degrades due to overshooting. As the 4^{th} order segmentation is a more conservative approach, these overshooting sections are compensated with the segment linearity, M_i. However, both 5^{th} order segmentations under-perform when compared to 3^{th} and 4^{th} order. This is because 5^{th}(a) order is more conservative than the 4^{th} order undershooting more, whereas 5^{th}(b) is more aggressive than the 3^{th} order producing larger overshoots. 0.3 and 0.4 Hz synthetic test results with an amplitude of 1 mV_{p-p} show that, 3^{th} order mean values are 16 µV, 26 µV and 10 µV, 12 µV less when compared to 2^{nd} and 5^{th}(a) order respectively.

Similarly, standard deviation RMS errors show that 3^{rd} order segmentation generates lower



(a) RMS mean error vs. frequency

(b) RMS standard deviation error vs. frequency

Figure 4.11.: Parametric segmentation analysis with 1 mV_{p-p} synthetic sinusoidal inputs and their fundamental frequency swept from 0.1 to 0.7 Hz.

Table 4.2.: WPL interpolation equations for 3 segments

# WPL Imp.	Parameter a & b values	Parameter c values	First Segment Eq. (H_{i_1})	Second Segment Eq. (H_{i_2})	Third Segment Eq. (H_{i_3})
WAM	$\frac{1}{2}, \frac{3}{4}, 1, ...\frac{5}{2}$	$\frac{3}{4}, 1, ...\frac{3}{2}$	$\frac{a * M_{i-1} + b * M_i}{a + b}$	$c * M_i$	$(3 - c) * M_i - H_{i_1}$

error variation when compared to conservative approaches such as 4^{th} and 5^{th}(a) orders. These higher orders are more in resemblance with linear interpolation and therefore, higher standard deviation is expected due to their nature of undershooting. However, as the sinusoidal frequency increases and the interpolation point sampling frequency remains constant, these higher order segmentation results improve. It can be seen that at around 0.6 Hz conservative approaches intersect with more aggressive ones.

Additionally, the computational complexity of the overall interpolation algorithm increases by two more additional conditions and multiplications per sample, as the segmentation order gets incremented. Therefore, based on the overall results and the computational complexity requirements, a $3rd$ order segmentation implementation proves to be a better approach. However, in applications where interpolation point sampling can not be guaranteed, a more conservative approach with additional computational load can still be implemented.

4.5.3. WPL Interpolation Equations

Following segmentation analysis, parametric tests on WPL interpolation equations have been carried out. Similar to the previous analyses, tests have been carried on 1 mV_{p-p} synthetic sinusoidal inputs with fundamental frequencies ranging from 0.1 up to 0.7 Hz. Table 4.2 shows the general expression of Eq. 4.10 for each segment.

Initially, tests have been performed on the second segment equation to define c parameter by using a single tone sinusoidal input and by sweeping the other two parameters, a and b. Fig. 4.12 and 4.13 show parametric test results of these tests with 0.3 Hz and 0.5 Hz sinusoidal input signals. The blue areas in each plot show lower RMS mean and standard deviation regions in all plots and c values of 1 and 1.25 provide best overall results for 0.3 Hz and 0.5 Hz sinusoidal input respectively. On the other hand, the difference between these two results is mainly correlated to the sampling frequency of the interpolation points and the input signal frequency. However, as these two results differ around approximately 3 µV in mean RMS and usually such a difference will be below noise floor, a computationally efficient approach is targeted and therefore, c parameter is defined as unity.

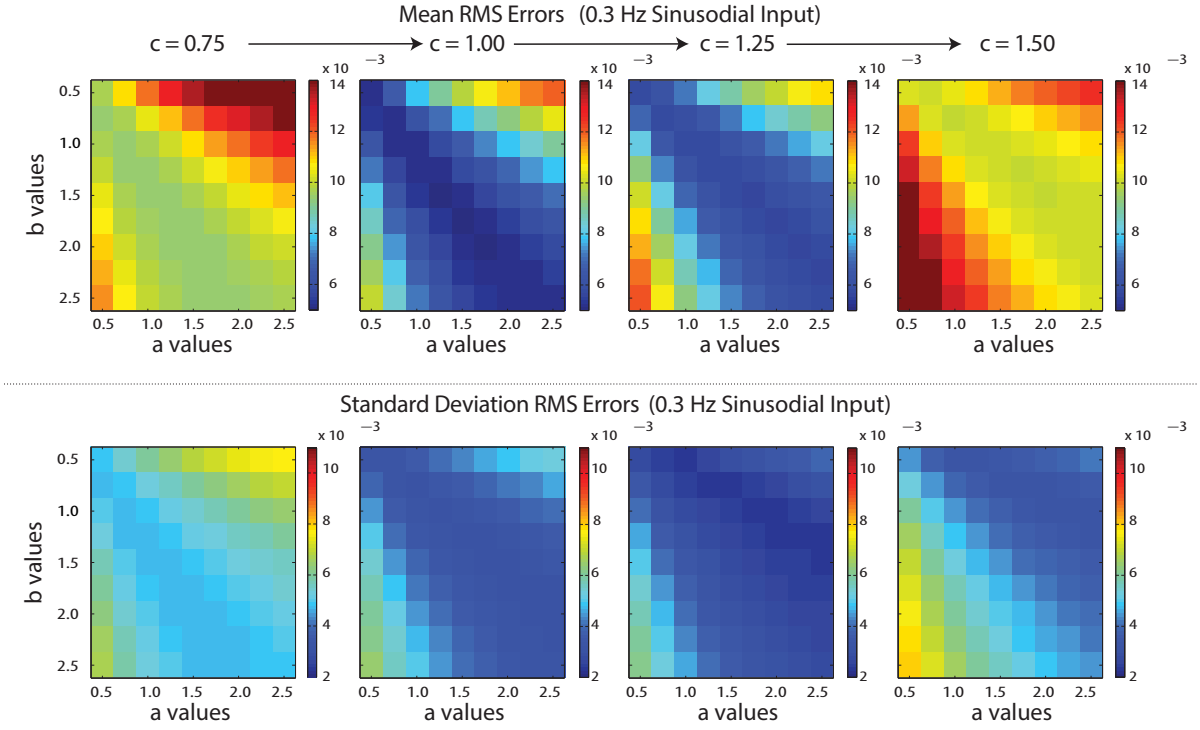


Figure 4.12.: Parametric analysis of coefficients, a, b and c, with 0.3 Hz sinusoidal input to determine WPL interpolation second segment equation, H_{i_2}.

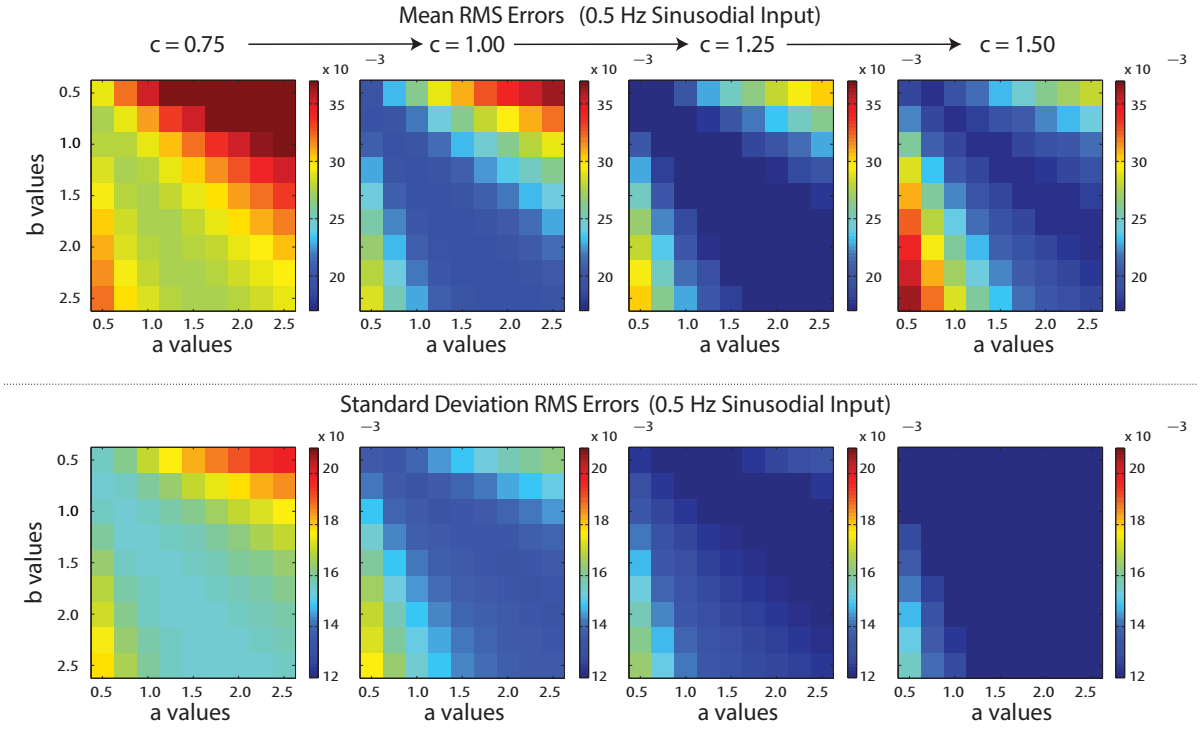


Figure 4.13.: Parametric analysis of coefficients, a, b and c, with 0.5 Hz sinusoidal input to determine WPL interpolation second segment equation, H_{i_2}.

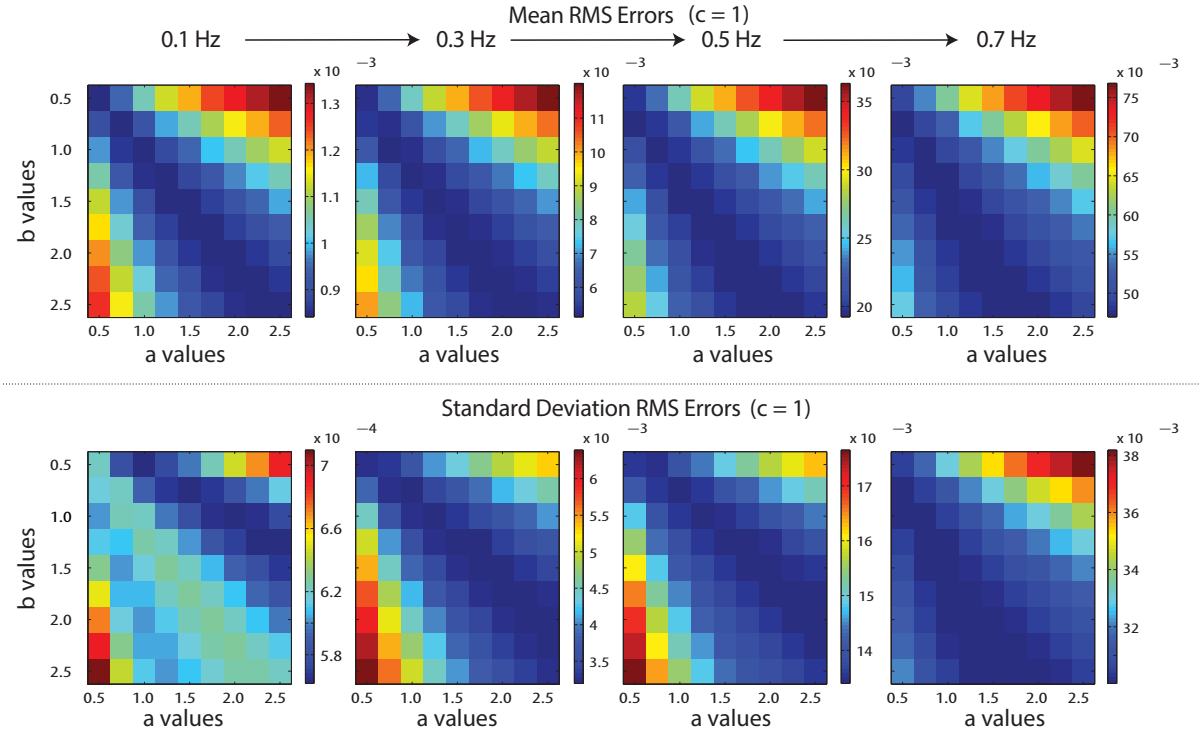


Figure 4.14.: WPL interpolation first segment equation, H_{i_1}, parametric analysis with sinusoidal inputs ranging from 0.1 Hz up to 0.7 Hz

Similar to the c parameter analysis, tests have been utilised on a and b parameters to define the first segment interpolation equation, H_{i_1}. This equation is based on the weighted arithmetic mean relationship of the previous and the current slopes namely, M_{i-1} and M_i. To investigate the relationship, parametric analysis tests have been performed and results with sinusoidal inputs and unity c parameter are plotted in Fig. 4.14. It can be seen that, RMS mean and standard deviation errors perform the same behaviour where the optimum point is defined diagonally in each plot requiring equality of parameters, a and b, at low frequency sinusoidal inputs. This relationship shifts towards larger b values with increased input frequency due to the constant interpolation point sampling frequency. Most of the a and b parameters generate good results; therefore, they are determined based on their computationally efficiency. Defining these parameters as unity shows that their accuracy responses are within blue shaded areas in all mean and standard deviation RMS error plots, and are affected least by input signal frequency as can be seen in Fig. 4.14.

Finally, the third segment is defined by the combination of these three variables and implemented as shown in Table 4.2. It has to be noted here that apart from weighted arithmetic mean average implementations, other mean relationships are also investigated. These tests include weighted geometric, w/harmonic, w/quadratic and w/heronian mean implementations. Apart from w/quadratic approach, other approaches generated worse results than weighted arithmetic mean implementation and required additional conditions to avoid divergence in baseline estimation since geometric and harmonic means are not defined for all real numbers. A similar

effect can be seen in PCHIP interpolation at instances where harmonic mean is undefined. In such cases the algorithm utilises pre-defined information as in Moler's work [168]. In our tested implementation, mathematically undefined conditions are checked with additional conditions and in its place weighted arithmetic mean implementation has been utilised at those undefined instances. On the other hand, w/quadratic results are approximately equal to the weighted mean arithmetic results. However, it is computationally more demanding requiring 4 additional conditions & multiplication operations per interpolation point. Supplementary figures and equations of these approaches are provided in Appendix C.1.

4.6. Results & Discussion

Once the design parameters are set, a theoretical error analysis is carried out to test our algorithm, and these results are then validated using synthetic and real sets of data comparing our results with other algorithms. As discussed in the previous chapter in more detail, the baseline wander signal can be modelled as a sinusoid around 0.15 - 0.3 Hz frequency with a varying fundamental frequency depending on exercise [40]. This information will be forming the basis of theoretical analysis and synthetic test data.

4.6.1. Theoretical Analysis

In this section, theoretical error analyses of WPL and linear interpolation in sinusoid estimation have been carried out algebraically. As interpolation points vary in time and depend on various factors, assumptions based on heart morphology have been applied in order to reduce the number of unknown parameters. Complete list of these employed assumptions are itemised below:

- Three coordinates are required during WPL interpolation namely (x_{i-2}, y_{i-2}), (x_{i-1}, y_{i-1}) and (x_i, y_i) to generate the previous slope, M_{i-1} and the current slope M_i. These interpolation points are detected after P, T waves and QRS complexes within a single heartbeat. Therefore, the distance from x_{i-2} to x_{i-1} and x_{i-1} to x_i is determined by the heart morphology and heart rate variability (HRV). Typical values of P, T waves and QRS complexes are covered in Section 2.1.2 in more detail whereas in regards to R-R heart rate variability of adjacent heart beats, studies show root mean square successive difference of 10 to 30 ms [180–182]. In this study, however, the distances from x_{i-2} to x_{i-1} and x_{i-1} to x_i are assumed to be equal when expressing the weighted piecewise error equation to simplify the overall algebraic expressions. In reality, this is not the case due to physiological and emotional conditions.

- To evaluate the error functions, turning point detection needs to be defined for any two arbitrary points in time. It is expected to have 1 breath at least for every 3-4 heartbeats [183, 184]. This ensures minimum of 9 to 12 interpolation point detections by the fiducial point detection algorithm covered in the previous chapter. Therefore, the baseline wander period is divided into 8 equal segments ensuring at least 9 interpolation points

and the boundary conditions of the error function integrals are calculated from $\frac{\pi}{4}$ to $\frac{\pi}{2}$ with their Δ (difference of the two arbitrary points) defined from $\frac{\pi}{16}$ to $\frac{\pi}{4}$ to generate 3D images of the overall system. Therefore, WPL interpolation can be compared to linear interpolation at those sections.

- Theoretical segmental error functions, $Err_{WPL-S1/2/3}$, of the WPL interpolation are calculated as in Eq. 4.14, where $f(x)$ denotes the real baseline wander and $f_{S1/S2/S3}(x)$ denotes the segmental baseline estimation. To determine the exact area under the curve, definite integral boundary conditions of each segment need to be determined based on the location of zeros of each segmental error function, $f(x) - f_{S1/S2/S3}(x)$. Such an approach requires to solve for zero crossings of every two arbitrary points in time, whereas a simplified approach can be utilised to understand the general behaviour as in Eq. 4.15. This approach focuses on the divergence of the error functions by focusing on the absolute error of each segment and neglects the errors due to multiple zero crossings. As the interpolation algorithm is tracking the function accurately, these neglected errors are small when compared to the divergence error.

$$\begin{aligned} Err_{WPL-S1} &= \int_{x_1}^{x_{1_1}} f(x) - f_{S1}(x)\,dx \\ Err_{WPL-S2} &= \int_{x_{1_1}}^{x_{1_2}} f(x) - f_{S2}(x)\,dx \\ Err_{WPL-S3} &= \int_{x_{1_2}}^{x_2} f(x) - f_{S3}(x)\,dx \end{aligned} \tag{4.14}$$

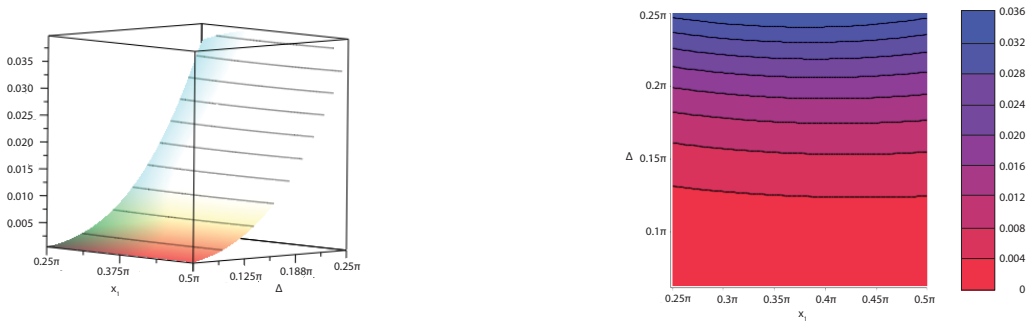
$$Err_{WPL} = \int_{x_1}^{x_{1_1}} |f(x) - f_{S1}(x)|\,dx + \int_{x_{1_1}}^{x_{1_2}} |f(x) - f_{S2}(x)|\,dx + \int_{x_{1_2}}^{x_2} |f(x) - f_{S3}(x)|\,dx \tag{4.15}$$

4.6.1.1. Analytical Data Test Results

Complete analytical expressions of both linear and weighted piecewise interpolation are shown in Appendix C.2. The differences between the analytical expressions of each interpolation method and the input sinusoids are denoted as the error and two types of plots have been presented to compare the accuracy of both interpolation methods. Even though the two plots are alternatives of each other, sometimes 3D plots cannot be self-explanatory, therefore contour plots can be used in assistance. For simplicity, the distance between interpolation points, x_2 and x_1, is abbreviated as Δ in all plots.

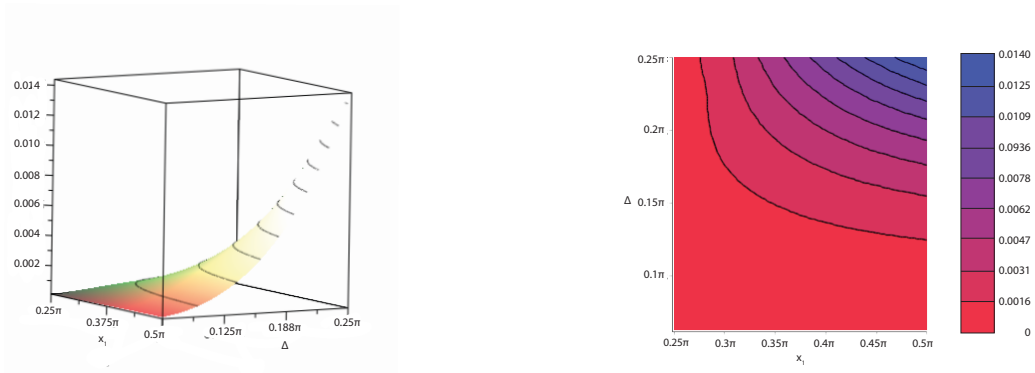
Linear Interpolation Numerical experiments have been run on sinusoids and error plots of linear interpolation with $sin(x)$ and $sin(2x)$ input signals are shown in Fig. 4.15 and 4.17.

Weighted Piecewise Linear Interpolation Similarly, numerical experiments have been carried out on WPL interpolation with sinusoidal inputs with different fundamental frequencies. Fig. 4.16 and 4.18 show the absolute error function of WPL interpolation as mentioned in Eq. 4.15 with $sin(x)$ and $sin(2x)$ input signals respectively. These tests have been utilised to quantify the theoretical improvement of WPL interpolation when compared to linear interpolation. On the other hand, Fig. 4.19, 4.20 and 4.21 show 3D error and contour plots of each segment separately as mentioned in Eq. 4.14.



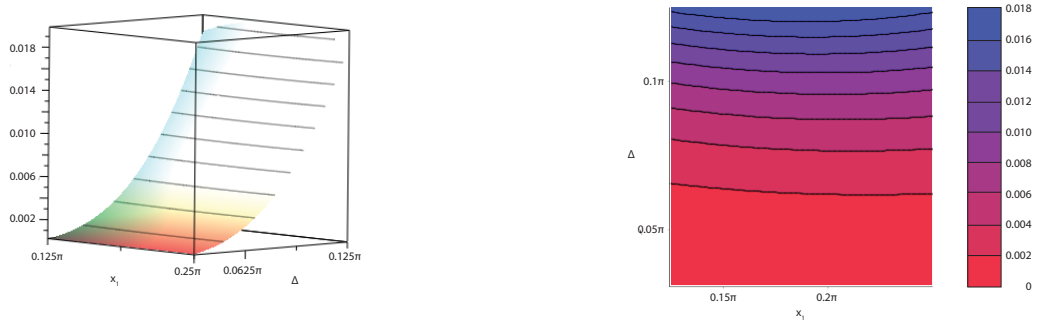
(a) Error 3D plot (b) Contour plot

Figure 4.15.: Error function plots of linear interpolation of $sin(x)$ at the interpolation point x_1 vs Δ



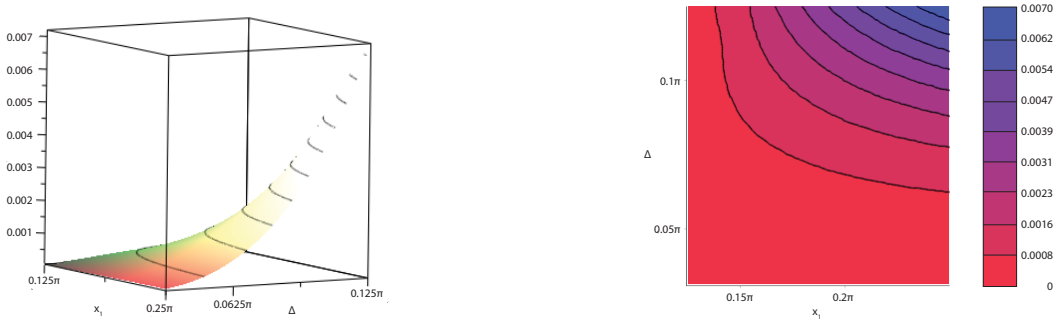
(a) Error 3D plot (b) Contour plot

Figure 4.16.: Absolute error function of WPL interpolation of $sin(x)$ at the interpolation point x_1 vs Δ



(a) Error 3D plot (b) Contour plot

Figure 4.17.: Error function plots of linear interpolation of $sin(2x)$ at the interpolation point x_1 vs Δ



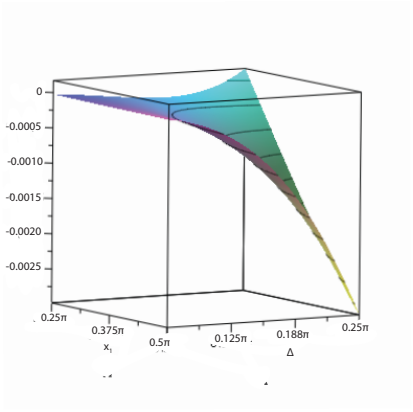
(a) Error 3D plot (b) Contour plot

Figure 4.18.: Absolute error function of WPL interpolation of $sin(2x)$ at the interpolation point x_1 vs Δ

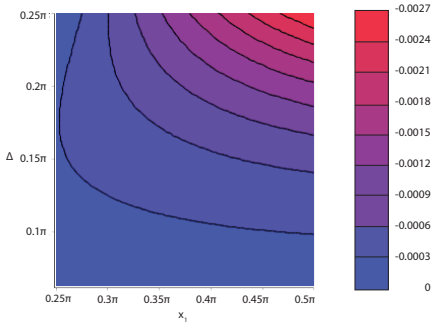
4.6.1.2. Analytical Data Test Discussion

Fig. 4.15 up to 4.18 show both interpolation methods for two different input frequencies. In both cases, WPL interpolation accuracy improvement is highly noticeable as can be seen in both 3D and contour plots. At the coordinates before the peak of the sinusoid, the algorithm is tracking the input signal accurately and the accuracy of the algorithm starts to degrade as the interpolation point approaches the peak. However, even in worst case scenarios where WPL interpolation overshoots at the peaks, the overall algorithm still performs much better than linear interpolation. These instances occur at the peaks when the distance between interpolation points x_1 and x_2 increases.

On the other hand, similar accuracy behaviour can be observed as the input frequency increases. The accuracy of the overall algorithm is related to the interpolation point sampling frequency in relation to the input signal frequency and when both double, better accuracy is expected as a consequence eventually. The improvement relates to the updated slope calculations

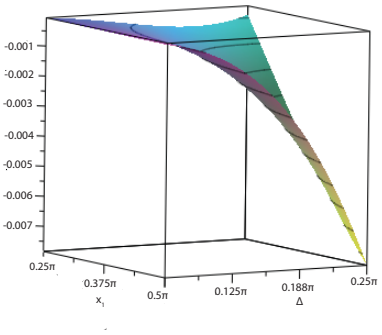


(a) Error 3D plot

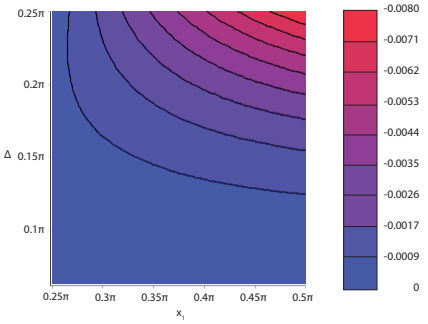


(b) Contour plot

Figure 4.19.: Error function plots of WPL interpolation (Segment 1 only) of $sin(x)$ at the interpolation point x_1 vs Δ

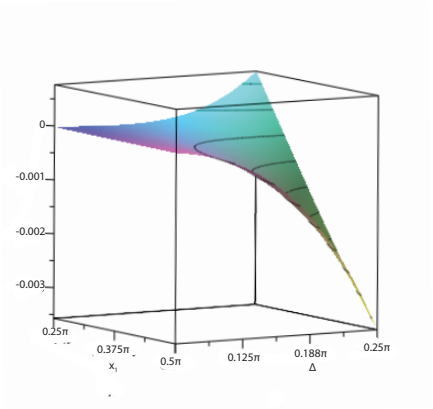


(a) Error 3D plot

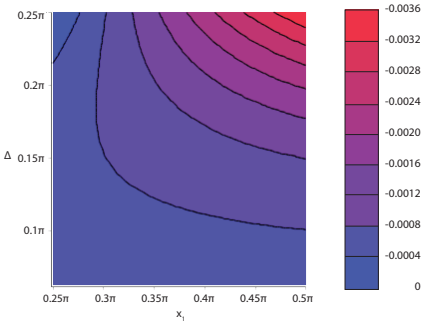


(b) Contour plot

Figure 4.20.: Error function plots of WPL interpolation (Segment 2 only) of $sin(x)$ at the interpolation point x_1 vs Δ



(a) Error 3D plot



(b) Contour plot

Figure 4.21.: Error function plots of WPL interpolation (Segment 3 only) of $sin(x)$ at the interpolation point x_1 vs Δ

and a better approximation occurs as in Runge-Kutta method. Fig. 4.17 and 4.18 show the 3D and contour error plots with input signal, $sin(2x)$. As can be seen, the error is exactly equal to the half of $sin(x)$ error plots as these only relate to the first half of the interpolation point introduced and do not include the updated slopes at remainder of the section when compared to $sin(x)$ input.

Additionally, plots of each segment show how well WPL interpolation is tracking the input signal and they are used to identify where both functions are converging or diverging. In other words, these plots can be utilised to identify the undershooting or overshooting conditions of the overall system. When the segmental analysis has been carried out as in Eq. 4.14, it is seen that at lower coordinates the error is almost close to zero. This means WPL algorithm is tracking the input signal accurately and undershooting error is compensated with overshooting at each segment as can be seen in Fig. 4.19, 4.20 and 4.21. When the distances between two arbitrary points increase and the interpolation point, x_1, approaches to the peak, the overall error function reaches to its maximum. This condition has been discussed in Section 4.5.1 in more detail and additional conditions have been utilised to overcome such instances; however, tests have shown that overshooting areas still perform better when compared to linear interpolation. Therefore, implementation has not been changed and the main reason degrading the performance of the WPL interpolation is thoroughly investigated and well known.

4.6.2. Synthetic Data Analysis

Synthetic data tests have been carried out in MATLAB as mentioned in Section 4.4. As the baseline wander can be modelled as a sinusoid around 0.15 - 0.3 Hz [40] and its frequency increases with exercise, sinusoids with their fundamental frequencies ranging from 0.05 Hz up to 0.7 Hz are utilised in this section. These frequencies correspond to approximately 3 to 42 breaths per minute respectively.

4.6.2.1. Synthetic Data Test Results

The results of WPL and other interpolation methods such as linear, cubic spline and piecewise cubic hermite interpolation (PCHIP) applied to synthetic data are evaluated. These tests are guided by 2243 heartbeats and the fiducial points belonging to the each heart beat are detected by the baseline detection algorithm. Fig. 4.22 shows RMS errors per heart beat of the synthetic data tested at various frequencies.

To investigate the mean and standard deviation variations of each algorithm, time domain responses are also plotted in Fig 4.23. Fig. 4.23(a) shows 0.3 Hz synthetic data with linear, cubic, WPL and PCHIP interpolation results whereas in Fig. 4.23(b) a more detailed time domain analysis have been shown comparing linear and WPL interpolation at two different frequencies, 0.3 Hz and 0.5 Hz, along with sample by sample error analysis.

4.6.2.2. Synthetic Data Test Discussion

During synthetic data tests, real interpolation points are generated from MIT-BIH Arrythmia Database signal 100m.mat file as mentioned in Section 4.4. This is because real interpolation points can vary in time due to respiration sinus arrhythmia (RSA) [185, 186]. In theoretical analysis, these details have been neglected to simplify analytical expressions and to investigate the general behaviour; however, during synthetic data tests RSA effects have been also included and it can be seen that WPL interpolation achieves more accurate results when compared to linear and PCHIP interpolation at all frequencies and results are almost comparable to the cubic spline errors as shown in Fig. 4.22. The same figure also shows that accuracy results depend on the input signal frequency in relation to the interpolation point sampling frequency. As the latter depends on the tested ECG signal recording, increasing the input signal frequency degrades the accuracy of all interpolation algorithms.

To express the results in the time domain, two discrete frequency (0.3 Hz and 0.5 Hz) responses are depicted in Fig. 4.23. As the heart rate of the patient is around 72 bpm, and given that respiration rate and pulse rate are related with a ratio of approximately 1 breath for every 3-4 heartbeats [183, 184], 0.7 Hz synthetic input results would correspond to 130 to 170 bpm. Therefore, 0.3 Hz and 0.5 Hz frequency responses are more realistic for algorithm evaluation based on the heart rate vs respiration rate relationship. Time domain results and sample by sample error analysis at these frequencies are depicted in Fig. 4.23. Fig. 4.23(a) shows the results of 4 different algorithms namely, linear, PCHIP, cubic spline and WPL interpolation. As can be seen from the figure near the 3 second mark, linear interpolation generates the

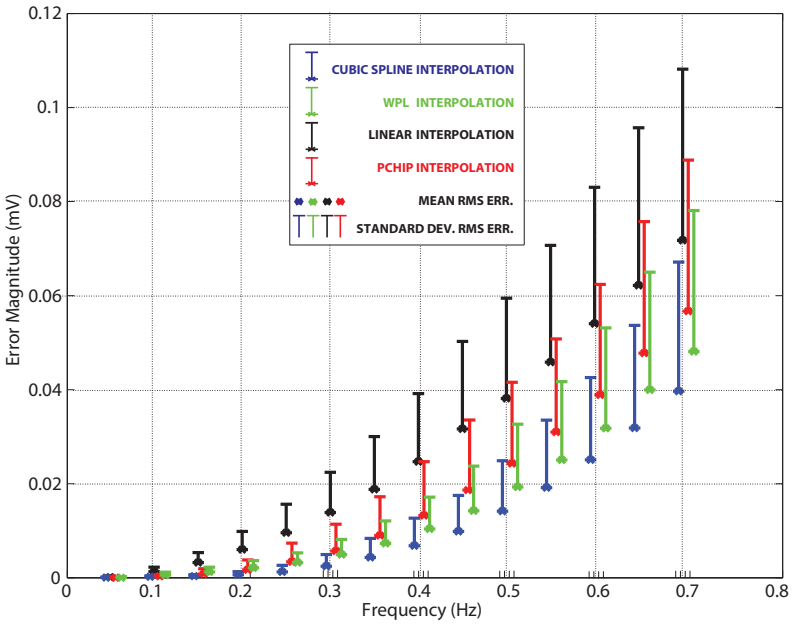
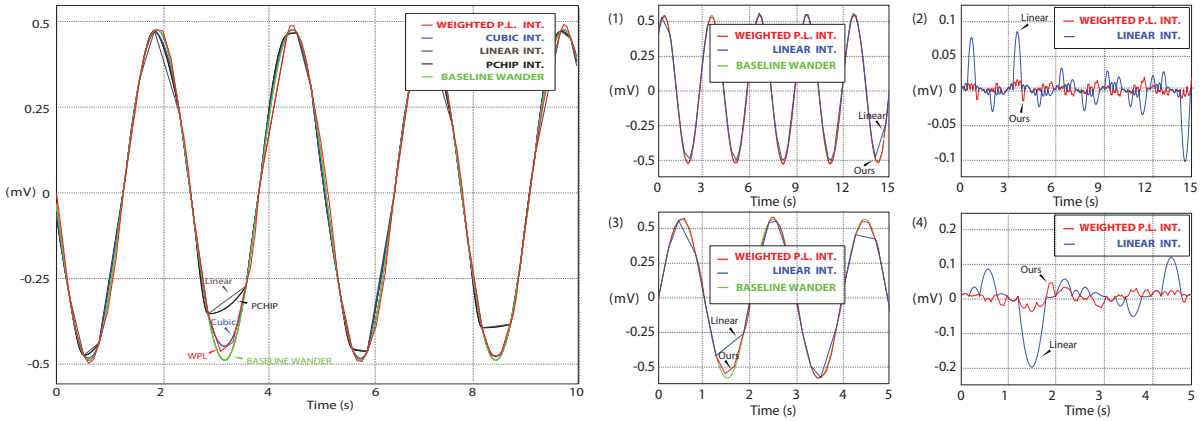


Figure 4.22.: Synthetic data test results of interpolation algorithms showing mean & standard deviation of RMS errors per heart beat



(a) 0.3 Hz sinusodial input and time domain error analysis. Signals are denoted as: Baseline wander (Green); Linear int. (Brown); WPL int. (Red); Cubic int. (Blue); PCHIP int. (Black)

(b) WPL vs linear interpolation: (1) 0.3 Hz sinusoid response; (2) sample by sample error analysis; (3) 0.5 Hz sinusoid response; (4) sample by sample error analysis

Figure 4.23.: Comparison of WPL interpolation with other algorithms using a $1\,\mathrm{mV}_{p-p}$ sinusoidal signal at 0.3 Hz and 0.5 Hz input signal

worst results among all interpolation algorithms whereas PCHIP estimate is well undershooting when compared to the original signal. On the other hand, cubic spline and WPL interpolation estimates are the closest estimates. Even though, the algorithm does not perform better than the cubic spline approach, WPL interpolation is less complex and it does not require any windowing techniques as well as triangular matrix solving to calculate the coefficients.

4.6.3. Real Data Analysis

Following analytical and synthetic data test results, MATLAB tests with recorded data have been executed and evaluated. To do so, recorded baseline wander datasets (BWM1.mat and BWM2.mat) are acquired from the MIT-BIH Noise Stress Database [152] and details of those datasets and pre-processing applied on these data sets are covered in the previous chapter. As mentioned before, these pre-processing methods aim to eliminate the white noise present in these recordings and reduce the white noise below 5 µV in worst conditions such that noise floor is not defined by this random noise and interpolation methods can be tested thoroughly.

4.6.3.1. Real Data Test Results

Similar to the tests on synthetic data, four different interpolation algorithms (Linear, Cubic Spline, PCHIP and WPL) are tested in MATLAB and their results with mean, median and standard deviation RMS errors per heart beat and maximum absolute error per ST segment are shown in Table 4.3. These results are also generated by utilising same real interpolation points that are acquired from MIT-BIH Arrythmia Database signal, 100m.mat, over 2243 heartbeats along with the annotation files used to define ST segments. The graphical representation of Table 4.3 results is shown in Fig.4.24 with histogram plots to asses the probability distribution

of continuous error functions. In these plots three quantities have been focused showing RMS error per heart beat and ST segment as well as the maximum absolute error per ST segment.

While evaluating the cubic spline and PCHIP interpolation results, windowing techniques have been utilised to avoid dependency on the past data or monotonicity requirements of the interpolation methods respectively. Real baseline wander estimates of linear and WPL interpolation in the time domain along with their sample by sample error analysis results are shown in Fig. 4.25 and 4.26. These time domain results show the error analysis on both BWM1.mat and BWM2.mat datasets, which are realistic baseline wander recordings acquired from the MIT-BIH Noise Stress Database.

4.6.3.2. Real Data Test Discussion

Four different interpolation algorithms have been tested as indicated in the results section. The general behaviour of real data test results acquired from MATLAB simulations is matching to the low frequency synthetic data test results. Similar to these tests, linear interpolation acts as the worst algorithm in real data tests and cubic spline approach generates the most accurate results, whereas WPL interpolation is comparable in accuracy and more preferable than its polynomial counterparts due to its simplicity. On the other hand, histogram results in Fig. 4.24 show that linear and PCHIP interpolation error distributions are more spread, while error distributions of WPL and cubic spline interpolation are similar with reduced number of counted large errors (above 50 µV) than linear interpolation results. This occurrence is also mentioned in synthetic data tests and the main reason of such a spread distribution both in linear and PCHIP interpolation is due to the undershooting instances at curvature points.

Table 4.3.: Real data - RMS and maximum absolute error per heartbeat and ST segment

Interpolation Method	Signal (Hz)	RMS error (µV) per heartbeat†			Max. Abs. error (µV) per ST segment†		
		μ	median	σ	μ	median	σ
Linear Interpolation	BWM1	14.8	10.6	13.1	28.8	21.7	25.1
	BWM2	8.4	7.1	5.5	16.2	14.5	9.7
Cubic Spline Interpolation	BWM1	13.5	9.2	14.2	26.1	19.3	21.8
(Windowed N=3)	BWM2	7.9	6.4	5.4	15.3	13.6	8.0
PCHIP Interpolation	BWM1	13.5	9.5	13.3	26.2	19.5	21.7
(Windowed N=3)	BWM2	8.0	6.7	5.6	15.5	13.9	8.2
WPL Interpolation	BWM1	13.7	10.0	12.7	26.8	19.8	22.1
	BWM2	8.1	6.9	5.2	15.5	13.7	8.6

†2243 Heartbeats detected via MIT-BIH Arrythmia Database (100m.mat)

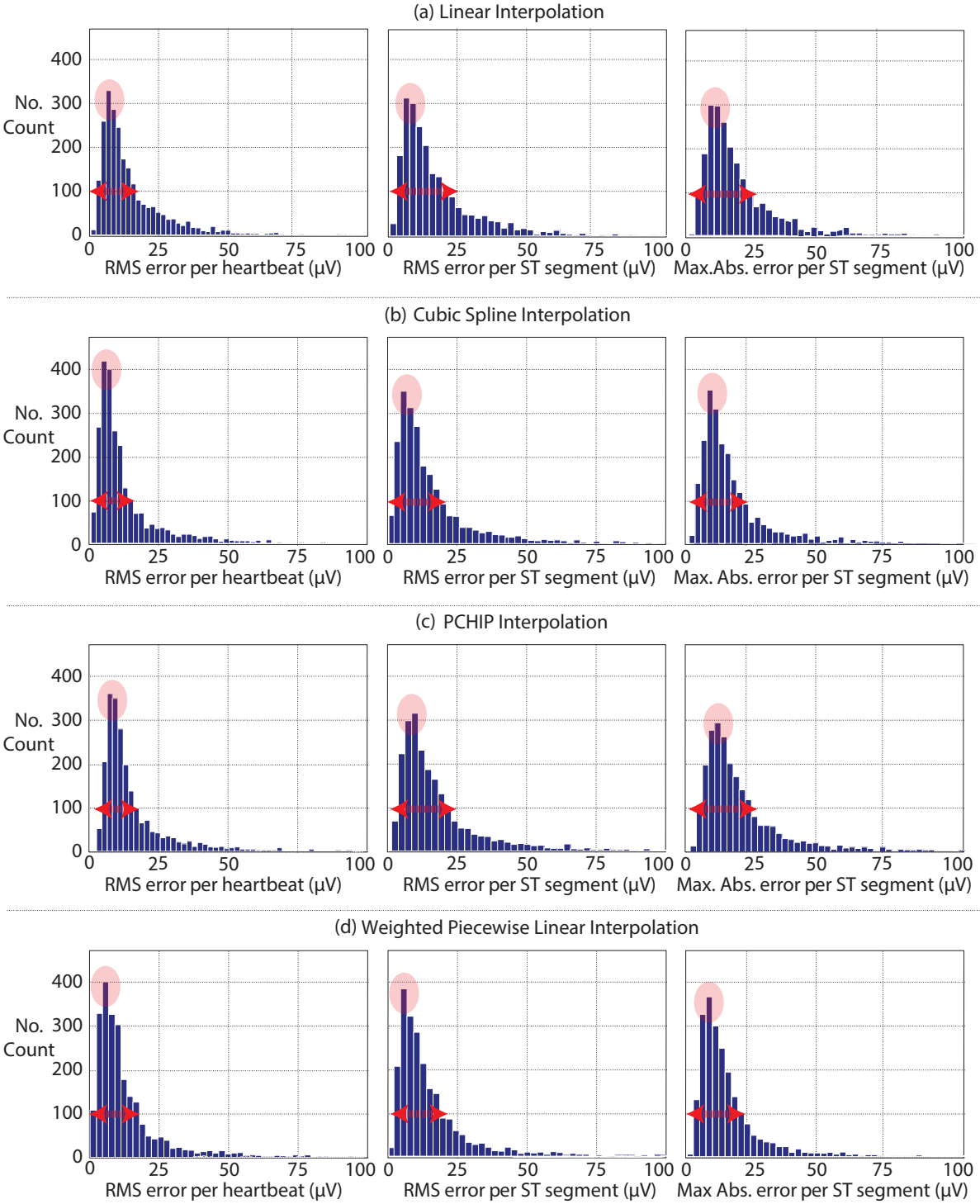


Figure 4.24.: Real baseline wander signal, BWM1.mat, RMS and maximum absolute error per heart beat/ST segment histogram results. Shown are for: (a)linear interpolation; (b) cubic spline interpolation; (c) PCHIP interpolation; (d) WPL interpolation

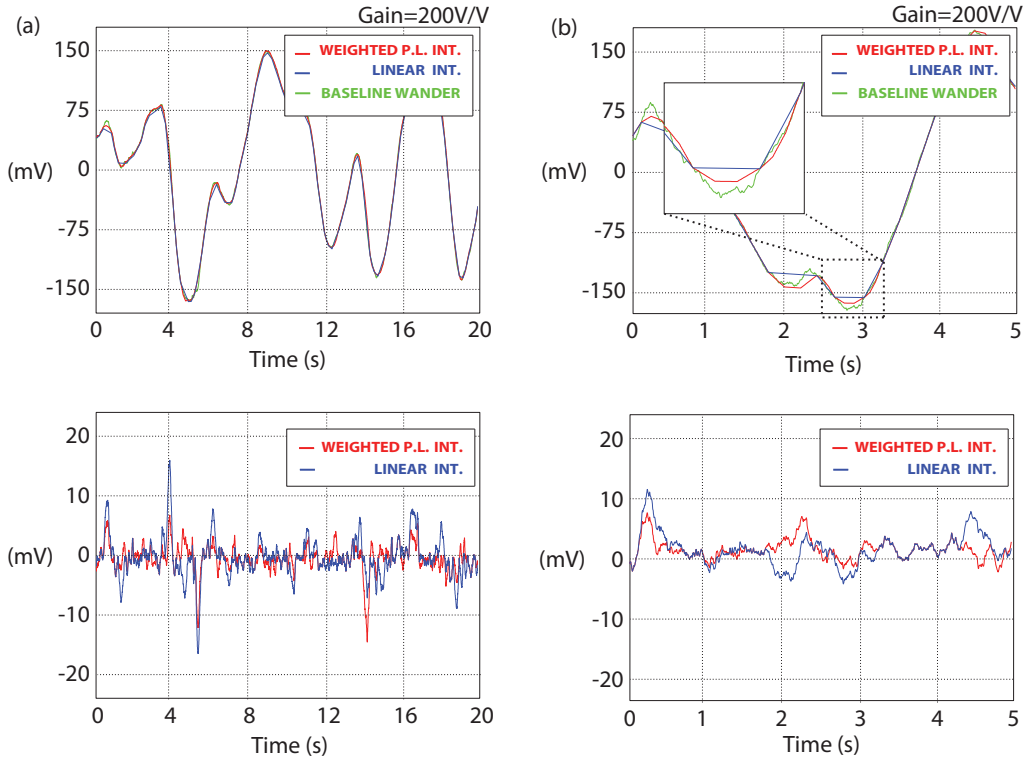


Figure 4.25.: Linear and WPL interpolation comparison with real baseline wander signals: (a) BWM1.mat input signal; (b) Its sample by sample error analysis; (c) BWM2.mat input signal; (d) Its sample by sample error analysis

Time domain responses of linear and WPL interpolation in Fig. 4.25(b) show that improvements at peaks and valleys can be achieved at various instances. Error analysis of both Fig. 4.25(a) and (b) plots express that WPL interpolation results are more accurate. An occasional overshooting, however, might occur as in the BWM1.mat subplot. It should be also noted that accuracy improvements are subject to Nyquist sampling rate limitations such that high frequency content cannot be recovered by any interpolation algorithm. This condition is clearly shown in Fig 4.26. In this example, a 0.12 Hz respiration signal with residual Gaussian noise generates error results comparable to the one at 0.4 Hz respiration rate. As the impedance seen by the amplifier changes and even though low frequency content error estimations generate more accurate results, due to the white noise present on the signal the accuracy improvement is limited on this occasion. Therefore, not all of the results in Table 4.3 are related to systematic interpolation errors.

Table 4.3 results show that there is a large variation in BWM1.mat and BWM2.mat results. These differences are due to higher standard deviation (93 vs 36) and higher kurtosis (15.6 vs 4.3) of BWM1 signal when compared to BWM2 signal. In other words, BWM1 signal variation in amplitude and peakedness is higher; therefore, accuracy performance degrades in all interpolation algorithms. Possible causes of such difference can be related to gender differences, stress test conditions or even lung capacity of the patient as the impedance change seen by the amplifier can drastically alter these test signals. In the event of missing fiducial

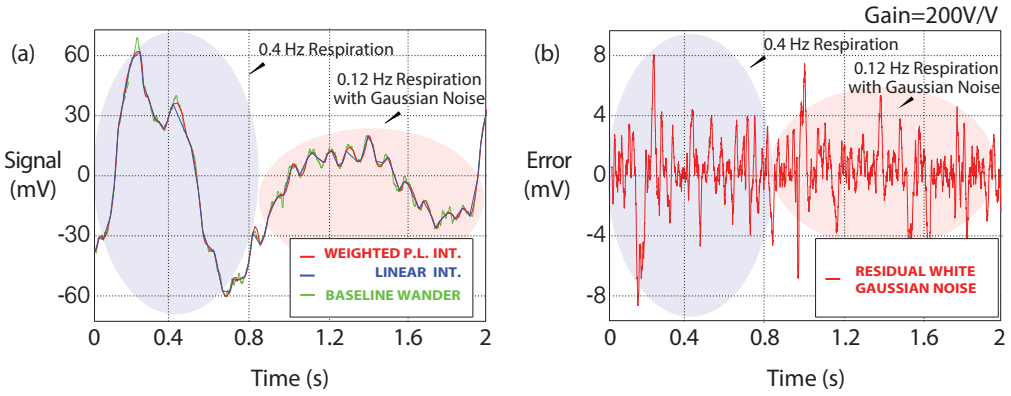


Figure 4.26.: Effect of residual Gaussian noise on linear and WPL interpolation algorithm. Signals are denoted as real baseline wander (green), linear interp. (blue), WPL interp. (red) Shown are: (a) BWM1.mat response; (b) WPL interpolation sample by sample error analysis

point detections on the other hand, RMS and absolute errors increase. These results are also observed in synthetic data tests where increasing the input signal frequency without changing the interpolation point sampling degraded the accuracy results of every algorithm.

In evaluating both of these test signals, not only RMS but also maximum absolute errors are calculated since ST segment carries additional information about the patient. Even though RMS results carry a good measure of the real effect of sinusoids like baseline wander, maximum absolute errors at ST segments are noticeably higher as expected. As the American Heart Association (AHA) and International Electrotechnical Commission (IEC) standards allow a maximum error of 100 µV for clinical ECG systems during ST segments [17–20], lower mean and standard deviations results are crucial while preserving the system complexity. For this reason, WPL interpolation provides the best trade off between accuracy and complexity and forms the basis of our interpolation algorithm design.

4.7. Conclusion

Various interpolation algorithms exist in the literature and most of these approaches are based on polynomial estimations. This chapter has proposed a novel interpolation algorithm utilised in baseline estimation within the context of accuracy requirements defined by the American Heart Association (AHA) and International Electrotechnical Commission (IEC) standards.

The proposed algorithm has merits of a hybrid approach, focusing on improving accuracy and reducing computational complexity. Turning point sections are preserved by generating comparable results to its polynomial counterparts, and computational complexity requirements are reduced where possible. This way, the developed algorithm can be implemented on low-power hardware.

It has been shown that ST segment distortion with the WPL interpolation is comparable to the presented higher order polynomial interpolation techniques. Real data tests convey an RMSD and a maximum absolute error of 13.7 µV mean, 10.0 µV median with 12.7 µV standard

deviation, and 26.8 µV mean, 19.8 µV median with 22.1 µV standard deviation on the BWM1 signal acquired from the MIT-BIH Noise Stress Database, respectively. When these errors are compared to cubic spline interpolation, less than 1 µV mean, median RMS errors are observed per heart beat and per ST segment. According to these results, WPL interpolation exhibits comparable accuracy with less computational complexity as opposed to its polynomial counterpart. Compared to linear interpolation, undershooting instances are minimised, which shows an accuracy improvement in the maximum absolute mean and median errors observed in the ST segment by more than 2 µV.

When the histogram plots and the time domain responses of each interpolation technique are closely investigated, cubic spline and WPL interpolation exhibit a more condensed distribution compared to PCHIP and linear interpolation due to undershooting instances as presented in their time domain responses. As for their computational requirements, WPL interpolation requires less hardware resources when compared to polynomial counterparts. Therefore, it is preferable on low-power hardware implementation systems, leaving enough headroom for the overall system to estimate the baseline wander accurately while preserving the signal integrity.

Chapter 5

ECG Baseline Drift Removal In Low Power Real-Time Hardware

Real-time ECG hardware systems have existed in the market for a long period of time. However, most of these systems often distort the signal of interest as they utilise AC coupling which limits the accuracy of baseline wander removal as discussed in Chapter 2. Next generation systems, on the other hand, aspire to achieve real-time noise interference removal implementations while preserving the signal integrity. The feasibility of these systems relies on the efficiency of the real-time algorithms and the potential of their low-cost, low-power, and low-area requirements.

With the advent of technology, the cost of individual components such as microcontrollers (MCUs), high resolution analogue-to-digital converters (ADCs) and instrumentation amplifiers (IAs) required in a real-time system implementations is becoming more affordable and is often easily accessible as these are manufactured by the most well-known companies such as Texas Instruments Corporation, Analog Devices and many others.

Today, the bottleneck in real-time system designs is often due to the high computational complexity of baseline wander detection algorithms. As covered in Section 2.2.3, most algorithms utilise iterative runs with multiplication operations and require extensive data storage limiting their implementation in digital signal processors (DSPs) and microcontrollers. Their real-time adaptations, on the other hand, utilise windowing techniques and due to limited data storage available in processing, accuracy degradation is inevitable in such approaches.

In this chapter, the embedded system implementation of the proposed algorithms described in Chapter 3 and Chapter 4 is presented in detail. In Section 5.4, the C implementation of the overall algorithm and its memory requirements are described. In Section 5.5, the overall algorithm is implemented on a Texas Instruments' MSP430 microcontroller unit (MCU), and the baseline wander removal results are compared with simulated results. In addition, the total number of instructions per cycle required by each stage of the baseline wander estimation and the interpolation algorithms are quantified and the energy efficiency of their implementations are presented.

5.1. Objectives

Focusing on the real-time hardware-efficient implementation, the key objectives of this chapter can be summarised as follows:

- **Accuracy:** The proposed low-computational complexity algorithms developed in MATLAB have to be adapted to C and the built-in functions utilised in MATLAB need to be replaced with their real-time C counter-parts. In addition, the default precision format in MATLAB is double-precision. However, C implementation and MCU responses should be implemented in single-precision format to reduce power and area and the potential degradation in performance due to single precision needs to be quantified.
- **Resource Utilisation:** Real-time implementations are restricted by the latency requirements based on the available amount of memory and MCU instructions in relation to the clock frequency. Each sample is required to be processed within a certain period of time and buffers are required to guarantee that no information is lost during processing. Therefore, available resources required by the developed algorithms need to be quantified such that real-time baseline wander estimation is targeted while preserving the signal integrity.
- **Embedded System Implementation:** The overall algorithm needs to be tested on the microcontroller as in real-time implementation and the errors involved with transmission and its overall effect on the baseline estimation need to be quantified. Finally, as the target application aims for ambulatory design, the total power consumption of the overall system has to be investigated.

5.2. Background

In this chapter, the embedded system realisation of the developed algorithms is investigated. As the target application aims for ambulatory design, hardware interfaces such as ADC, and communication interfaces such as Serial Peripheral Interface (SPI) and Universal Asynchronous Receiver/Transmitter (UART) are a necessity to test the microcontroller implementation.

In terms of battery life, higher clock frequency increases the overall current consumption; therefore, a mid-range system clock with low power dissipation increases the battery life. Finally, the architecture type determines the tested application and 16-bit and 32-bit microcontrollers can both be a viable option when the variables are represented with single precision.

Multiplication operations require a high number of instruction per cycles; therefore microcontrollers with dedicated hardware multipliers are investigated to reduce the total number of instructions per cycle required by the overall system.

TI launchpads provide one of the least active current consumption at 100 µA / MHz with a clock frequency operating at 16 MHz (16-bit) - 48 MHz (32-bit) when compared to other manufacturers. These devices have UART and SPI connections, a dedicated hardware multiplier, and a 12-bit ADC with 128 kB RAM storage.

5.3. Challenges

Even though developed algorithms are computationally efficient, their implementation in real-time embedded systems carries challenges. These challenges require identification of each physical parameter of the MCU to accurately estimate the baseline wander while preserving the signal integrity, and can be listed as follows:

- **Precision:** Developed algorithms and their real-time implementations are required to maintain accuracy results defined by the standards and preserve signal integrity in baseline estimation. Single-precision implementations of the developed algorithms and their effect on the overall system performance need to be quantified to achieve a viable embedded system implementation.
- **Latency:** Even though developed algorithms avoid multiplication operations as much as possible, the IIR and FIR filter implementations require 32-bit floating point calculations. Therefore, the required number of instructions per cycle of the overall algorithm should match the MCU capabilities.
- **Accuracy:** Errors in relation to transmission operation, clock frequency generation and clock skew might distort the signal of interest. In addition, any type of data transmission introduces mis-read/transmitted bits occasionally and degrades the system performance. The resulting effect of these systematic errors needs to be quantified and identified.
- **Run-time Operation:** In ambulatory operations battery power is crucial and reducing the total number of instructions per cycle increases the run-time of the battery cycle. For this reason, utilisation of hardware multipliers needs to be investigated to reduce the system clock frequency and power consumption consequently.

5.4. C implementation

This section covers the adaptation of developed algorithms into C environment and involves any built-in functions utilised in MATLAB simulations to be adjusted into their real-time representations in C. The simulations are done using GNU Compiler Collection (GCC) compiler using the Xcode (Version 6.4) development environment. In addition, tests are done utilising single-precision floating-point format which occupies 4 bytes (32-bits) in computer memory. This format, namely referred to as IEEE 754 standard, can express all integers with six or fewer significant decimal digits without loss of information in addition to some integers up to nine significant digits [187].

5.4.1. Biquad Filtering Implementation

In digital signal processing, a biquad filter is a second-order recursive linear filter with two poles and zeros. The gain of such a filter is denoted with G in Eq. 5.1, whereas numerator coefficients,

$b_0,\ b_1$ and b_2, define the feed-forward path and the denominator coefficients, $a_0,\ a_1$ and a_2, form the feedback path of a biquad filter implementation.

$$H(z) = G * \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \tag{5.1}$$

In digital signal processing, biquad filters are often utilised as building blocks to avoid unstable operation as higher order implementations are sensitive to coefficient accuracy.

In baseline wander estimation, the second-order sections (SOS) matrix generated in MATLAB is converted into transfer function form by utilising the built-in function, *sos2tf*. Subsequently, the filter stage of the developed algorithm has been tested with the built-in function, *filter*, in the MATLAB environment. This function utilises numerator and denominator coefficients in double precision and generates an accurate and stable filtering.

In the C implementation double precision of these filters requires computational resources both in memory and total number of instructions per cycle. For this reason, these filters are implemented in biquads as SOS forming a transposed direct-form-II implementation and single precision is used in defining the filter coefficients.

In fixed-point calculations direct-form I are often preferred as these topologies involve single summation points whereas in floating point calculations direct-form II implementations save two extra memory locations. The transposed topology of two second order recursive filter as shown in Fig. 5.1, has the same filter characteristics whereas the intermediate sums are achieved with close-valued numbers achieving higher precision.

Filter coefficients are determined as covered in Section 3.5.2 using *fdatool* in MATLAB. These coefficients are then expressed in transposed direct-form II structure in single precision while their numerator coefficients are normalised. The other coefficients are then expressed with nine significant decimal digits in IEEE 754 binary format in Xcode and the response of these filters is then evaluated and compared with MATLAB results. The filter implementation in C is acquired using a single function for each biquad with pointers addressing the filter coefficients,

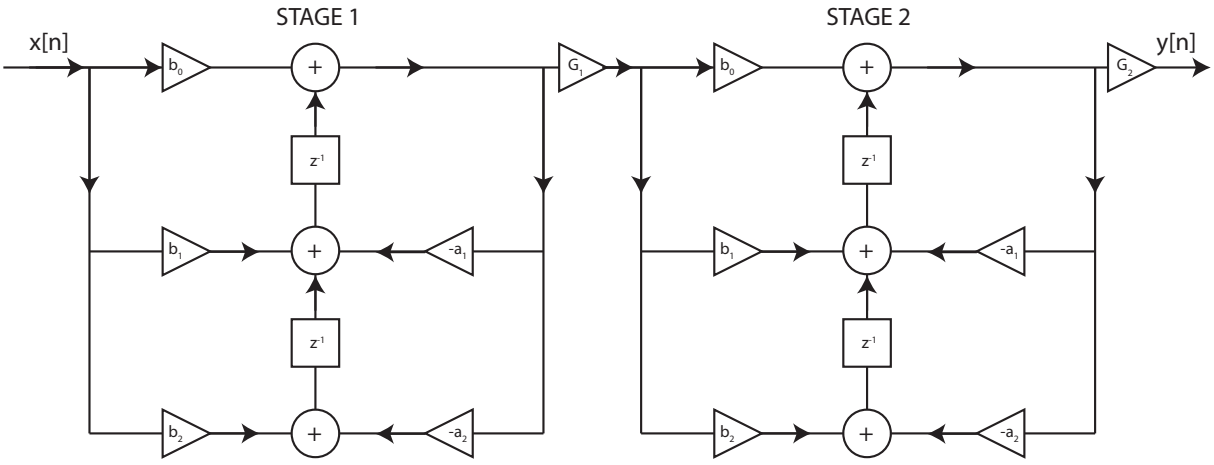


Figure 5.1.: Transposed-direct-form II implementation of two biquad (second order IIR) filters

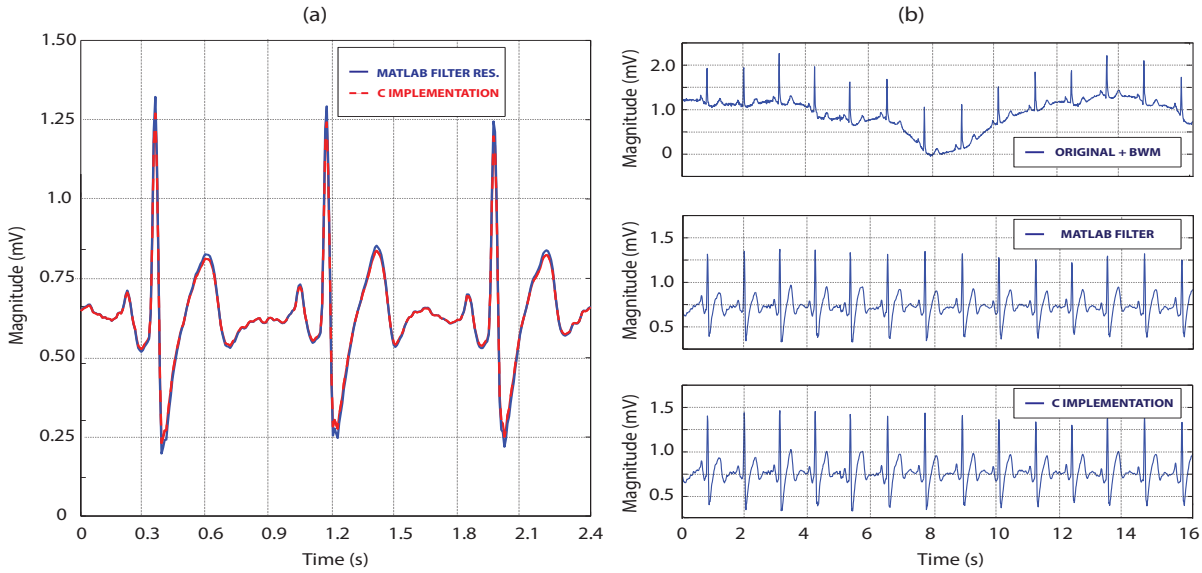


Figure 5.2.: Filter responses of MATLAB and C implementations. Shown are time domain responses of: (a) 3 heart beats with MATLAB and C implementation filtering plotted on the same graph; (b) Original signal with baseline wander, MATLAB filtering and single precision C implementation of the same filters

and the delay stage results that are statically stored in the memory. The complete function of the real-time filter implementation is provided in Appendix D.

Fig. 5.2 shows MATLAB and C filter responses with double and single precision. The filter responses are stable and slight differences appear in the order of 10-20 µV deviation at the peaks. However, these deviations have no effect on fiducial point detections as the thresholds are determined after differentiation and moving integrator stages in a recursive method.

5.4.2. Interpolation Implementation

As covered in Chapter 3, various interpolation techniques are utilised in baseline wander estimation. These techniques use the built-in MATLAB interpolation function, *interp1*, which runs linear, cubic spline or PCHIP interpolation as the interpolation method. This function has been replaced with the computationally-efficient WPL interpolation algorithm introduced in Chapter 4 and its C implementation is provided in Appendix D.

The C-code before the interpolation stage runs in serial-in, serial-out format and depending on the method preferred, the interpolation algorithm generates the output either in serial-in, parallel-out or serial-in, serial-out format. Both implementation types require utilisation of buffers in order to avoid information loss.

In serial-in, serial-out output type, the algorithm requires fiducial point storage in the buffer. As these points are non-uniformly sampled, the duration between fiducial points might cause instability when no buffer is utilised. Such an instance occurs in events like slow or undetected heart beats. In these instances, the distance between two consecutive fiducial points increases

and this change stalls the program as the interpolation stage requires the upcoming fiducial point before it is detected. To avoid such cases, the C algorithm buffers fiducial points at start up and utilises the information stored in the buffer during these instances. A single heart beat delay is sufficient enough to overcome missing fiducial points. However, in the event of missing heart beats when the buffer is processed completely, the algorithm is put into a buffering process. This implementation can be seen in the interpolation section of the C coding in Appendix D. In dataset 101 of the MIT-BIH Arryhtmia Database with the baseline wander added from the MIT-BIH Noise Stress Database, the buffer stage is initialised only once over a 30 minute duration and that instance is shown in Fig. 5.3. For simplicity, discrepancy calculations are not shown in this plot and it can be seen that at the 1.8 second mark, the buffer is emptied and the interpolation output stalls for 300 samples whereas ideal interpolation at this instance is shown with green. After the re-initalisation, the algorithm recovers and continues as normal. These instances occur during large step changes and missing QRS complexes when no fiducial point is detected as can be seen in the plot.

On the other hand, a serial-in, parallel-out output implementation calls the interpolation function only when a new fiducial point is detected. Once a new interpolation point is located, the algorithm generates all the interpolated data and stores them in the buffer, which then can be pointed by the main function and subtracted. This implementation type increases the overall data storage and requires the processed data to be saved in the buffer to be fed back at the sampling rate for processing. Due to this extra storage requirement in the microcontroller implementation, a serial in, serial-out output type has been utilised.

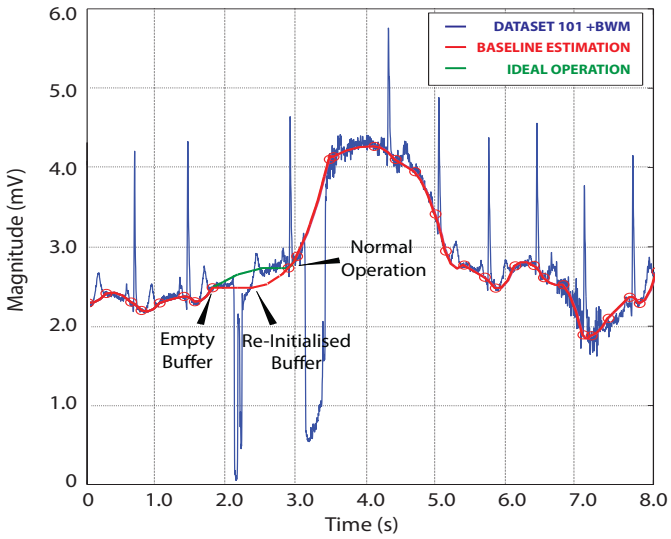


Figure 5.3.: Buffer re-initialisation of the interpolation algorithm

5.4.3. Computational Complexity

Power consumption and computational complexity might not be an issue for bedside systems as processing is often done by powerful computer platforms. However, when the target application aims for ambulatory system design, recording continuous data and utilising these platforms is not viable. Therefore, algorithms designed for such purpose require computational complexity quantification and profound analysis to assess their suitability for real-time operation.

Both algorithms covered in Chapter 3 and Chapter 4 have specifically targeted a real-time hardware implementation (unlike most algorithms mentioned in Chapter 2). The hardware complexity of both of these algorithms in terms of memory and computational requirements are therefore discussed in this section.

5.4.3.1. Memory

The vast majority of algorithms for baseline wander removal require a significant buffer (i.e. memory storage). The proposed algorithm, however, only requires memory for IIR filter coefficients, state variables for each stage and group delay compensation. Table 5.1 lists the required number of variables for algorithm implementation stage by stage.

The memory requirements of the first stage (S1) are determined by the filter implementations. The high-pass IIR filter implementation is stable using single precision and requires 10 coefficient values to be stored in the memory for a direct form 2-transposed implementation. Similarly, the low-pass cut-off frequency implementation is determined with a total of 10 coefficient values stored in the read-only memory, whereas moving average filtering is achieved with 4 coefficients.

The second stage (S2) requires storing a 5-point derivative, squaring and moving window integrator calculations in addition to the temporary variables required for system operation. It should be noted that the output of the squaring operation and the moving integrator can overflow when expressed with integers. Therefore, the output of these operations is required to be expressed by at least 32-bits (long) if they are expressed as integers and shifting operations are targeted.

The third stage (S3), on the other hand, is more complex. However, most of the flags and search parameters require only a single bit whereas wait and search windows can be expressed with eight bits.

Finally, the fourth stage (S4) requires a buffer, which stores the fiducial point locations and the generated slopes in addition to temporary variables. Eq. 5.2 shows the total memory allocation by the overall algorithm and each of its stages separately. The total memory required for baseline wander estimation is equal to 652 bytes in total.

$$N_{bytes} = 240\,(S1) + 60\,(S2) + 104\,(S3) + 212\,(S4) + 140\,(Main) = \ 652\ bytes \tag{5.2}$$

Table 5.1.: Memory requirements of each stage

Name	Symbol	Description	Type	Bytes
Filter Stage (S1)				
Static constants				
IIR Filter coefficients	a,b (3^{rd} order)	High-pass (f_H) coefficients	float	4×10
	a,b (3^{rd} order)	Low-pass (f_L) coefficients	float	4×10
FIR Filter coefficients	b (4^{th} order)	Moving av. coefficients	float	4×4
State variables				
ptr[12] + filter var[12]	ptr, var	Pointers and Filter variables	float	4×24
Delay Cells [12]	$(d1,d2)_{H,L}$, d_M	IIR filter delay cells	float	4×12
Pan & Tomp. Stage (S2)				
State variables				
i, ptr	i, ptr	Temporary Variables	int	2×2
y, sum	y, sum	Temporary Variables	float	4×2
Derivative [4]	$d_y d_t$	Derivative of the ECG signal	float	4×4
Integral [8]	$\int y d_t$	Integral of the ECG signal	float	4×8
Fiducial P.D. Stage (S3)				
State variables				
i, j, k, Count[2], WDT	i, j, k, C, WDT	Temporary Variables	int	2×6
Unfilt[5], Filt[2], MI[3]	U_f, F_f, MI	Un-/Filtered input	float	4×10
QRSflag	QRS_f	Derivative of the ECG signal	int	2
$Threshold_{QRS,P,T}$	T_{QRS}, T_P, T_T	QRS, P-/T- wave thresholds	float	4×3
P-/T- flag, search	P_f, P_s, T_f, T_s	P-/T- flag & search variables	int8	1×4
Wait, search window	W_w, S_w	Fiducial p. search variables	int8	1×2
F_{loc}[3]	$F_{1,2,3}$	Fiducial point locations	float	4×3
F_{diff}	J1,J2,J3	Discrepancy variables	float	4×3
F_{diff}	(J1J2), (J2J3)	Fiducial point discrepancies	float	4×2
WPL Interp. Stage (S4)				
State variables				
i, delay, Count, $temp_x$	i, d, C, t_x	Temporary Variables	int	2×4
$temp_+$, $temp_-$ $temp_y$	t_+,t_-,t_y	Temporary Variables	float	4×3
Slope1[12], Slope2[12]	S_1, S_2	Slope buffers	float	4×24
Duration[12], Locy[12]	D, L_y	Duration & location buffers	float	4×24
Main				
State variables				
$Input_{o,f,m,ix,iy}$	$Input_{o,f,m,ix,iy}$	Temporary Variables	float	4×12
$Output_{1,2,3}$	$Output_{1,2,3}$	Temporary Variables	float	4×3
Buffer	$Filt_i$, $UnFilt_i$	Buffer for filtered	float	4×20

5.4.3.2. Computation

In this section, an approximate measure to system complexity is targeted as the exact evaluation of the overall algorithm is hard to achieve and depends on various parameters. The algorithm is split into different functional blocks to quantify the number of operations and Table 5.2 shows complexity requirements of each stage based on basic ALU instructions in total number of addition/subtraction, multiplication/division and condition operations as well as memory requirements.

The total effective number of computations is determined by the number of computations per sample, $C_{per\ sample}$, and per interpolation (fiducial) point, $C_{per\ interpolation\ point}$ as in Eq. 5.3. Eq. 5.4 shows the effect of interpolation point generation in relation to sampling frequency, I_G, on the overall system complexity.

$$C_{Total} = C_{per\ sample} + I_G * C_{per\ interpolation\ point} \tag{5.3}$$

$$I_G = \frac{T_{interpolation\ point}}{T_{sample}} \tag{5.4}$$

The filtering stage processes every downsampled sample, therefore, its complexity measure is straightforward to calculate. The conditions in the filtering stage are determined by the states of its biquad implementation, and the total number of operations are independent of these state conditions. The moving average filter, on the other hand, can be implemented with shifting operations, however this requires truncating the output of the IIR filters (as shifting operation can be achieved on integers only). As long as real-time operation is satisfied, these operations are handled with full precision.

The Pan & Tompkins stage, similar to the filtering stage, processes every downsampled sample. Ideally, the derivative and integrator calculations can be handled with shifting operations as the generated numbers are quite large and less susceptible to noise, however similar to the filtering stage, no truncation is performed in this stage.

During fiducial point estimation, QRS detection, threshold generation and watchdog operation every downsampled sample is processed however, the total number of operations in regards to fiducial point detection depends on various factors such as multiple threshold detections, missing P-/T- waves, and isoelectric discrepancy compensation. Table 5.2 does not constitute these instances and shows the total number of operations based on fiducial point detection, M. An approximate quantification of fiducial point detection instances can be achieved based on the heart rate and characteristics (P, QRS and T waves). Under normal conditions, the resting heart rate for adults is substantially lower than 100 beats a minute [188], [189]. Even though it is not easy to estimate an individual's heart rate precisely, a patient with a constant heart rate of 72 bpm sampled at 360 Hz is expected to generate an interpolation point every 100 samples ($M = \frac{1}{100}$). This number can increase with lower heart rates, or decrease vice versa. Of course,

there are other factors affecting interpolation point quantification such as the duration of each heart segment or fluctuations on heart rate variability (HRV); however, these parameters are neglected in complexity quantification for all algorithms.

Discrepancy calculations, on the other hand, are done at the start up for 8 consecutive heart beats and these calculated discrepancies do not change unless a significant difference is detected or the algorithm is re-initiated. Therefore, their effect on the overall effective complexity is negligible.

Finally, the WPL interpolation stage is computationally non-exhaustive compared to higher order interpolation approaches as this technique does not rely on past information storage like higher order polynomial approaches and the number of total multiplication operations is limited. In this manner, this approach aims to bridge the gap between complexity measure and accuracy as these two factors usually appear as a trade off. Table 5.2 shows the complexity measure based on WPL interpolation and does not constitute the instances which are acquired with linear

Table 5.2.: Baseline wander estimation algorithm computation complexity per sample

Stage	Memory Access	Conditions	Add. & Subtract	Multiply & Divide
Filter Stage (S1)				
IIR Filtering (HPF)	28	1	8	10
IIR Filtering (LPF)	28	1	8	10
FIR Filtering (MAF)	10	-	3	4
Pan & Tomp. Stage (S2)				
5-point Derivative	4	-	3	3
Squaring	-	-	-	1
Moving Integral	1	1	2	2
Fiducial P.D. Stage (S3)				
QRS Flag & Watchdog	4	5	-	-
Threshold Generation	4	6	1	5
QRS Detection	10*M	3+M	1	-
T Wave Detection	6*M	3+M	1	1
P Wave Detection	4*M	3+M	-	1
Fiducial Point Detection	36*M	18*M	M	M
Discrepancy Calculations	14*N	10*N	N	2*N
WPL Interp. Stage (S4)				
Slope + Buffer	10+48*M	8+M	5	2
WPL Interpolation	18	12	6	8

interpolation. When compared to polynomial approaches, cubic spline interpolation requires fourteen floating point multiplications, ten additions and three conditions and an evaluation of an $N \times N$ triangular matrix, where N is defined by the window size of the real-time cubic spline interpolation [190]. This N by N matrix solution maintains the continuity of the overall system by evaluating the second derivatives of the interpolating function at the interpolation points and such an approach is computationally exhaustive.

As the polynomial approaches preserve the continuity of the interpolation function, baseline estimation is often contaminated with white noise and the accuracy results do not show an effective improvement as discussed in Chapter 4. Therefore, WPL interpolation is preferred due to its computationally effective advancements when compared to polynomial approaches.

5.5. MCU

As indicated in Section 5.2, certain MCU characteristics are looked for in determining the embedded system realisation. Based on low power consumption TI MSP430FR6989 launchpad series are determined as the choice for implementation. These 16-bit MCUs have active current consumption of 100 µA/MHz, and 350 nA at standby with real-time clock. Maximum clock frequency is defined at 16 MHz and three type of clocks are provided (ACLK, MCLK and SMCLK). In addition, the peripherals offered by this launchpad enable the communication with a MATLAB test platform through a serial universal asynchronous receiver/transmitter (UART) connection.

The overall system realisation is shown in Fig. 5.4. Oscilloscope (LeCroy WavePro 7300A) is utilised to affirm the clock frequency of the MCU and Code Composer Studio (CCS) v6 is used as the integrated development environment (IDE) to develop an interrupt service routine (ISR) in system testing.

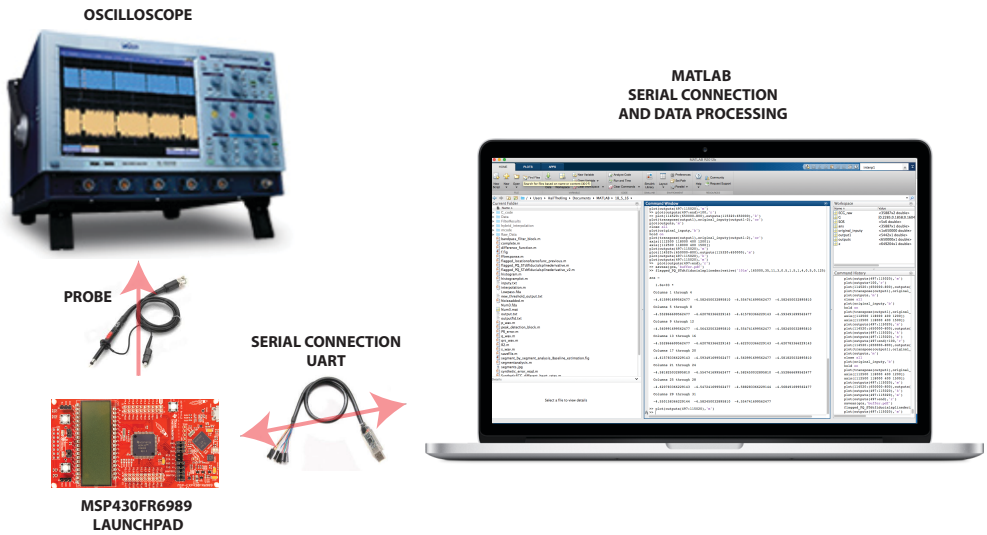


Figure 5.4.: Testing on an embedded system - MSP430FR6989

5.5.1. System Clock

The system clock defines the maximum number of instructions per cycle that can be utilised to process a sampled signal without loss of information. For this reason, it has a direct impact on the total power consumption of the microcontroller in addition to the peripherals utilised throughout system design. MSP430FR6989 launchpad provides an option to set the register of the MCU to utilise three system clocks to select best balance of performance and power consumption. These system clocks utilise low frequency, high frequency, or digitally controlled internal oscillators with dividers to achieve various clock frequencies up to 16 MHz.

In the embedded system realisation, Table 5.2 results are accounted for setting the system and peripheral clocks, MCLK and SMCLK. As presented in that table, approximately 50 multiplication operations per sample are required, and the system clock is initiated at 16 MHz. This way, the suitability of the system clock frequency in terms of total number of instructions per cycle is tested. It should be noted that clock frequencies above 8 MHz, exceed the ferroelectric random access memory (FRAM) access time, and therefore a waitstate is required. The register settings for the clock and the waitstate configuration are provided in Appendix E.

5.5.2. Peripherals

5.5.2.1. UART

Utilisation of certain peripherals is required in testing of the system realisation of the baseline wander estimation algorithm. One of these peripherals involves data communication between the MSP430 and the MATLAB environment to evaluate the accuracy of the processed data. This data transfer is achieved via the utilisation of the universal asynchronous receiver/transmitter (UART) protocol.

During the serial data communication, the first UART transmits a byte as individual bits, which are then re-assembled by the second UART back into a byte. Due to this operation, a string of binary code is generated and its data framing depends on the application type [191]. In Fig. 5.5, the data framing utilised for ECG baseline wander estimation on MSP430FR6989 launchpad is presented. This data frame utilises a total of 10-bits which consists of a start bit, 8 data bits, and a stop bit. A parity bit can be included in data transfer to detect errors in communication and the incorrect data can then be discarded.

The transfer rate is determined by the baud rate set by the UART in each device. In cases

BIT NUMBER	1	2	3	4	5	6	7	8	9	10
	START BIT 1	DATA BIT 1	DATA BIT 2	DATA BIT 3	DATA BIT 4	DATA BIT 5	DATA BIT 6	DATA BIT 7	DATA BIT 8	STOP BIT 1

Figure 5.5.: Utilised UART data frame on MSP430FR6989

where the baud rates are not matching, the data can be either misinterpreted or missed; therefore, both devices require the same speed and the utilisation of data ports to transmit data successfully. The total time spent for each byte transfer is expressed as in Eq 5.5 where baud rate is denoted as BR and the total number of bits sent in each cycle is represented as NB.

$$T_{per\ byte} = \frac{BR}{NB} \tag{5.5}$$

Each complete cycle transmits or receives a byte through the UART communication protocol in bits. Therefore, the speed required to receive and transmit the data must be sufficient enough to process ECG data, which requires 6 complete cycles per sample. The detailed breakdown of such a requirement originates from the 16-bit input data represented as an integer and the processed 32-bit output data represented as float. For a typical baud rate of 9600 bits per second, the total time elapsed for data communication is 6.25 ms, which is slower compared to the sampling frequency. In a complete system analysis, the time spent for data communication needs to be counted as a part of allowable instructions per cycle to preserve signal integrity, meaning that higher baud rates are required. Fig. 5.6 shows the baud rate and the maximum allowable instruction per cycle relationship of the overall system based on 6 bytes of data transfer sampled at 360 Hz for different system clock frequencies.

Based on the standard baud rates and typical SMCLK frequencies, timing errors are expected in terms of the sum of individual bit timings. To reduce the cumulative bit error, modulation features of the baud rate generator are utilised and registers of the UART configuration are set to minimise these errors. For a clock frequency of 16 MHz and a baud rate of 230400, a float representation is transmitted in 170 µs and a maximum transmit/receive error of 1.36 µs and 3 µs is expected based on the recommended baud rate settings [192]. The detailed UART register code of MSP430FR6989 utilised in baseline wander estimation is included in Appendix E.

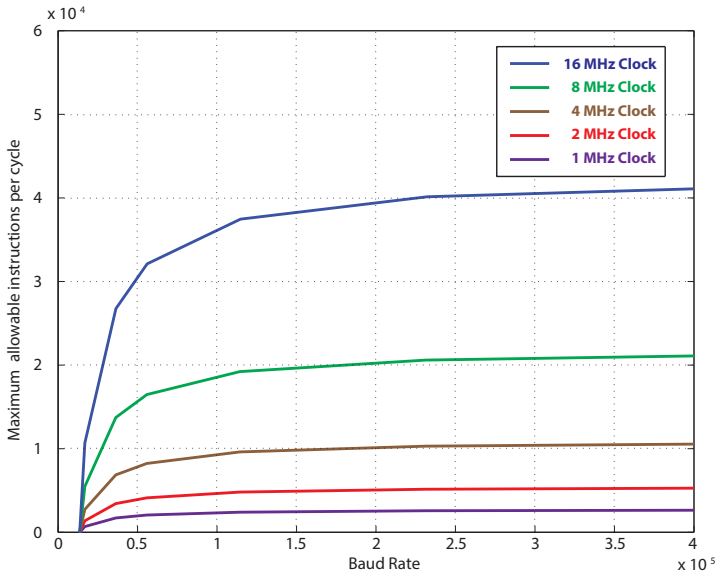


Figure 5.6.: Maximum allowable instructions per cycle vs baud rate

5.5.2.2. 32-bit Hardware Multiplier

Similar to the UART, a 32-bit hardware multiplier (MPY32) is a peripheral and its registers are loaded and read with CPU instructions. The hardware multiplier is able to achieve signed/unsigned multiply and accumulate operations with 8-bit, 16-bit, 24-bit and 32-bit operands.

The multiplication operation is started when the second operand is loaded into the registry and the result is generated within a certain number of clock cycles depending on the operation as specified by the datasheet [192]. In the saturation mode, 32-bit operation requires 11 MCLK cycles after OP2L is written. Therefore, delay cycles are needed to guarantee a successful multiplication operation, before reading data from the 16-bit result registers (RES0, RES1, RES2, RES3). The code generated to initiate the multiplier control registry written for 32-bit hardware multiplier is provided in the CCS Code in Appendix E.

In the absence of a hardware multiplier, MSP430 provides approximations based on Horner's method [193]. This approach requires the multiplier and the divisor to be known in advance. Therefore, such an approximation cannot be utilised on unknown variables. There are also other existing methods reported in the literature [194]. Due to the absolute error introduced in float operations, these methods, however, are not utilised in baseline wander estimation.

5.6. Embedded System Test Results

In this section, real data test results presented in Table 3.4 are validated with C implementation and MSP430 test results and presented in Table 5.3 and Table 5.4 respectively. These tests involve both of the algorithms developed in Chapter 3 and Chapter 4. The main motivation in presenting both C implementation and MSP430 results is to evaluate the true system response of both algorithms without any built-in functions. These results are then compared to embedded system measurements to evaluate the accuracy of both algorithms and their implementation thoroughly.

System evaluation tests are performed as indicated in Section 3.4 by adding baseline wander signal, BWM1, at various SNR levels to the dataset 100 and 101 from the MIT-BIH Arrhythmia Database. Both C implementation and its embedded system response are evaluated based on the same evaluation metrics utilised for MATLAB tests as in Section 3.4.3. Table 5.3 and Table 5.4 results show that C implementation and embedded system results match when compared with each other.

When RMSD results in Table 5.4 are compared with the MATLAB real data results in Table 3.4, it can be seen that there is an average of 2.0 µV, 1.8 µV and 0.9 µV difference in mean, median and standard deviation respectively. However, these differences are expected as they are related to the interpolation method utilised in each approach. As indicated in Chapter 1, cubic spline interpolation polynomial performs better when compared to WPL interpolation. These differences, however, are negligible when compared to the maximum allowable errors defined by the standards whereas computational resource requirements of WPL interpolation are more relaxed when compared to cubic spline approach.

Table 5.3.: C implementation system realisation - RMSD errors of MIT-BIH Database signals with added baseline wander from MIT-BIH Noise Stress Database

Dataset (Real)	Att. (dB)	RMSD error (µV) (With motion art.)			RMSD error (µV) (Without motion art.)			Total beats	Int. Err
		μ	Med	σ	μ	Med	σ	#	ϵ
100+BWM1 (R)	0	43.3	30.5	67.2	33.6	29.8	18.2	2243	P-T
100+BWM1 (R)	6	25.8	20.3	32.5	23.5	20.2	14.4	2243	P-T
100+BWM1 (R)	12	20.6	16.6	22.8	19.5	16.6	12.6	2243	P-T
100+BWM1 (R)	18	18.6	15.2	19.5	17.9	15.1	12.0	2243	P-T
100+BWM1 (R)	24	18.0	14.7	17.7	17.4	14.7	11.5	2243	P-T
101+BWM1 (R)	0	46.2	32.0	84.5	34.9	30.9	18.5	1835	P-T
101+BWM1 (R)	6	33.0	23.6	58.6	26.9	23.2	15.3	1835	P-T
101+BWM1 (R)	12	28.8	20.4	56.7	23.3	20.2	13.5	1835	P-T
101+BWM1 (R)	18	27.5	19.7	56.4	22.2	19.4	13.0	1835	P-T
101+BWM1 (R)	24	27.1	19.4	56.4	21.9	19.3	12.8	1835	P-T
Average	-	**28.9**	**21.2**	**47.2**	**24.1**	**20.9**	**14.2**	**2039**	P-T

Table 5.4.: MSP430 system realisation - RMSD errors of MIT-BIH Database signals with added baseline wander from MIT-BIH Noise Stress Database

Dataset (Real)	Att. (dB)	RMSD error (µV) (With motion art.)			RMSD error (µV) (Without motion art.)			Total beats	Int. Err
		μ	Med	σ	μ	Med	σ	#	ϵ
100+BWM1 (R)	0	43.4	30.5	67.3	33.6	29.8	18.2	2243	P-T
100+BWM1 (R)	6	25.9	20.4	32.5	23.6	20.3	14.5	2243	P-T
100+BWM1 (R)	12	20.7	16.8	22.8	19.6	16.8	12.6	2243	P-T
100+BWM1 (R)	18	18.9	15.2	19.5	17.9	15.1	11.9	2243	P-T
100+BWM1 (R)	24	18.1	14.7	17.7	17.4	14.7	11.4	2243	P-T
101+BWM1 (R)	0	46.2	32.0	84.5	34.9	30.9	18.5	1835	P-T
101+BWM1 (R)	6	33.0	23.6	58.4	26.9	23.2	15.4	1835	P-T
101+BWM1 (R)	12	28.8	20.5	56.7	23.3	20.2	13.5	1835	P-T
101+BWM1 (R)	18	27.5	19.7	56.4	22.3	19.4	13.0	1835	P-T
101+BWM1 (R)	24	27.1	19.3	56.4	22.0	19.1	13.0	1835	P-T
Average	-	**29.0**	**21.3**	**47.2**	**24.2**	**21.0**	**14.2**	**2039**	P-T

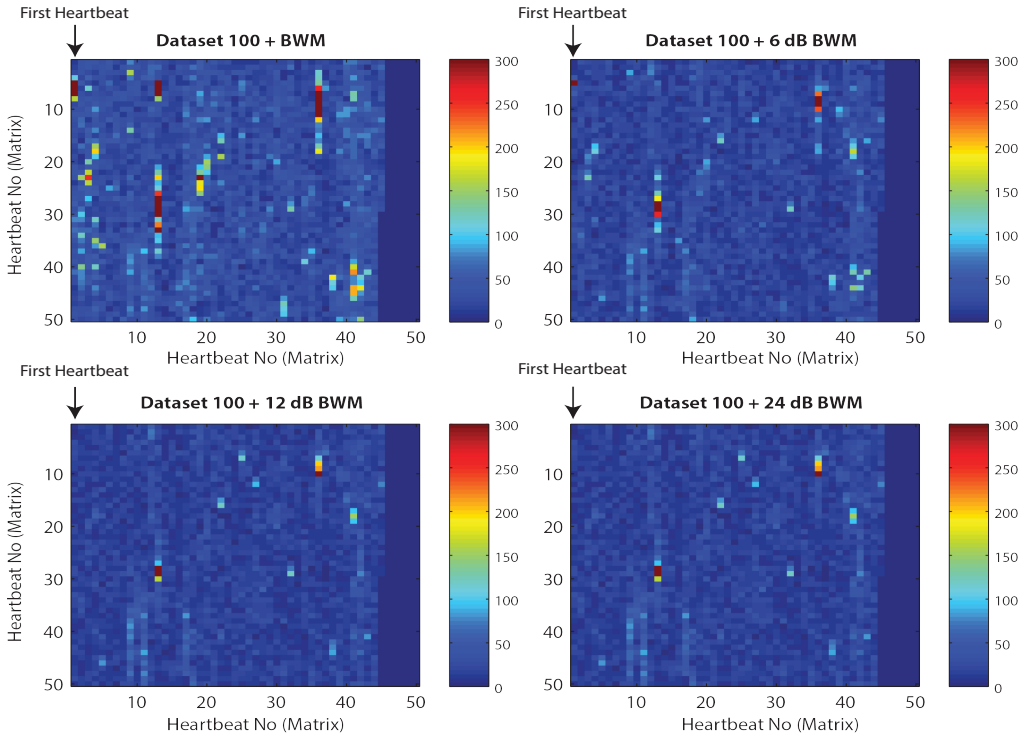


Figure 5.7.: MCU-based P-T interval heart beat error analysis of MIT-BIH Arrhythmia Database dataset 100 with added baseline wander, BWM1

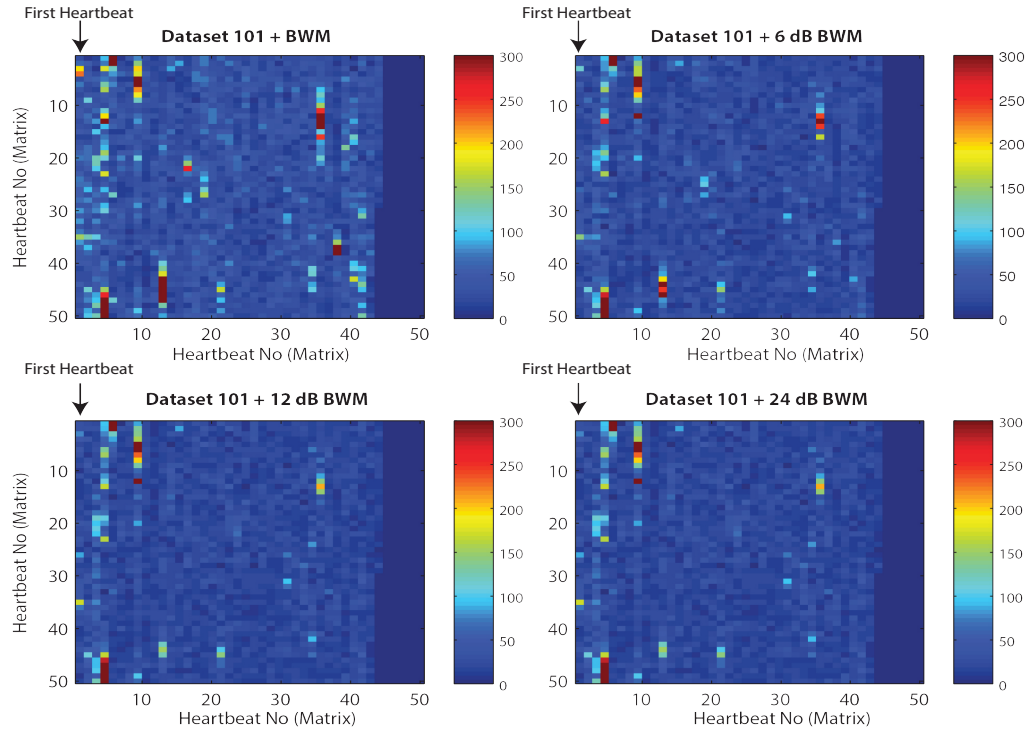
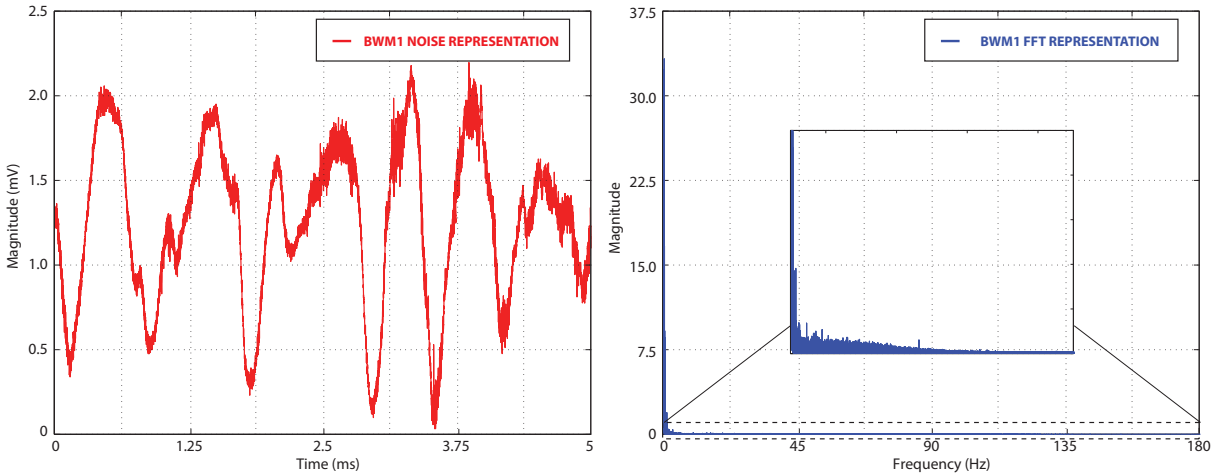


Figure 5.8.: MCU-based P-T interval heart beat error analysis of MIT-BIH Arrhythmia Database dataset 101 with added baseline wander, BWM1

Fig 5.7 and Fig 5.8 show the MCU-based P-T interval heart beat results of dataset 100 and dataset 101 with added baseline wander, BWM1, at various SNR levels. As the total number of heart beats in both datasets does not yield a good matrix distribution, missing heartbeats are filled with zeros to obtain a 50×50 matrix representation in these plots. As can be seen in both datasets with the BWM1 baseline signal attenuated by 24 dB, certain areas of the ECG recordings are contaminated. Their time domain response shows either EMG activity or step changes. When baseline estimation is utilised on these sections, the overall system accuracy degrades, independent of the utilised SNR level of the baseline wander. These noise artefacts define the noise floor, and the evaluation of the baseline wander estimation at these segments is not a realistic representation. However, as these noise artefacts subside, the overall algorithm recovers and baseline wander is detected accurately at the subsequent heart beats.

The same plots also show the effect of SNR levels and baseline wander degradation with increased noise. As indicated in Chapter 4, the baseline wander signal BWM1 shows a higher standard deviation (93 vs 36) and a higher kurtosis (15.6 vs 4.3); therefore, a 0 dB attenuated version of this noise source degrades the system performance by far the most. Specifically, at the start and at the end of the recording, the error results reach to higher levels. When the overall response is investigated, it is seen that these errors are due to white Gaussian noise present at certain sections of the baseline wander recording as can be seen in the time domain response in Fig 5.9(a). Similar to the EMG artefacts, when these errors define the noise floor, system degradation is expected as the high frequency content cannot be re-captured due to the Nyquist sampling theorem. When these datasets are utilised with a 6 dB attenuated version of the noise artefact, there is a substantial accuracy improvement on the whole dataset. Fast Fourier transform (FFT) of the BWM1 signal also confirms this observation as can be seen in Fig 5.9(b).



(a) Time domain response

(b) FFT response

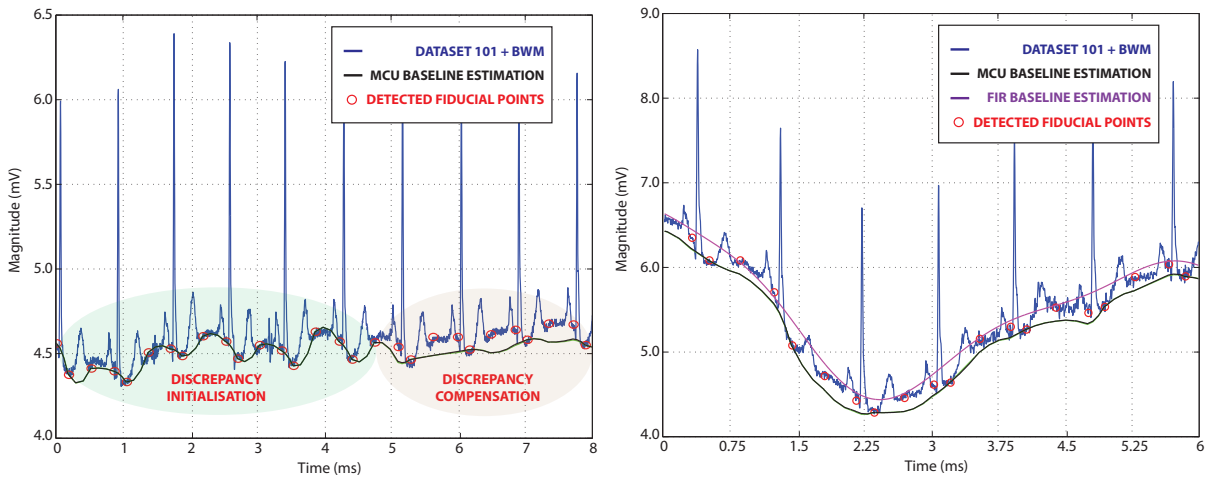
Figure 5.9.: Time domain and FFT response of MIT-BIH Noise Stress Database signal, BWM1

5.6.1. Embedded System Time-Domain Response

Throughout the thesis, it has been discussed that in reality fiducial points, *J1, J2* and *J3*, are not isoelectric. As covered in Section 3.5.5.1, this discrepancy information resides within the ECG signal and requires detection.

Fig. 5.10(a) shows the MCU response at the start-up to detect discrepancy differences between fiducial points. As can be seen, initially the baseline estimation passes through every single detected fiducial point for 8 consecutive successful heart cycles until discrepancy information is stabilised. After that, upcoming fiducial points, *J1 and J3*, are adjusted according to the calculated information to achieve a more realistic baseline wander estimation. It should be noted that discrepancy information is introduced on the *J1 and J3* fiducial points only to preserve *J2* level at all instances.

The baseline estimation with initialised discrepancy calculations is shown in Fig. 5.10(b). In this plot, FIR filter response mentioned in Section 3.4.3 is also depicted with the purple graph. This filter avoids non-linear phase distortion and the deformation due to ringing is minimal as the presented signal does not contain any step changes. When these distortions are kept to a minimum, the FIR filter detects baseline wander content below 0.67 Hz accurately and its response is in strong resemblance with the MCU baseline estimation with discrepancy calculations. During baseline estimation, however, if the discrepancy calculations are not included, the baseline estimation passes through every single detected fiducial point, resulting in heart-related information to be removed. Such an approach degrades the accuracy of the overall algorithm and the degradation magnitude depends on the discrepancy information residing within the signal. When evaluating dataset 100 and 101 with 24 dB attenuated BWM1 signal as in Table 5.3 and Table 5.4, tests without introducing discrepancy information at the start-up resulted in 31.2 µV mean, 28.9 µV median, 12.0 µV standard deviation, and 39.1 µV mean, 37.7 µV median, 12.3 µV standard deviation RMSD errors respectively. As can be seen, discrepancy information



(a) Discrepancy initialisation

(b) Discrepancy compensated baseline estimation

Figure 5.10.: MCU-based baseline estimation with discrepancy initialisation and compensation

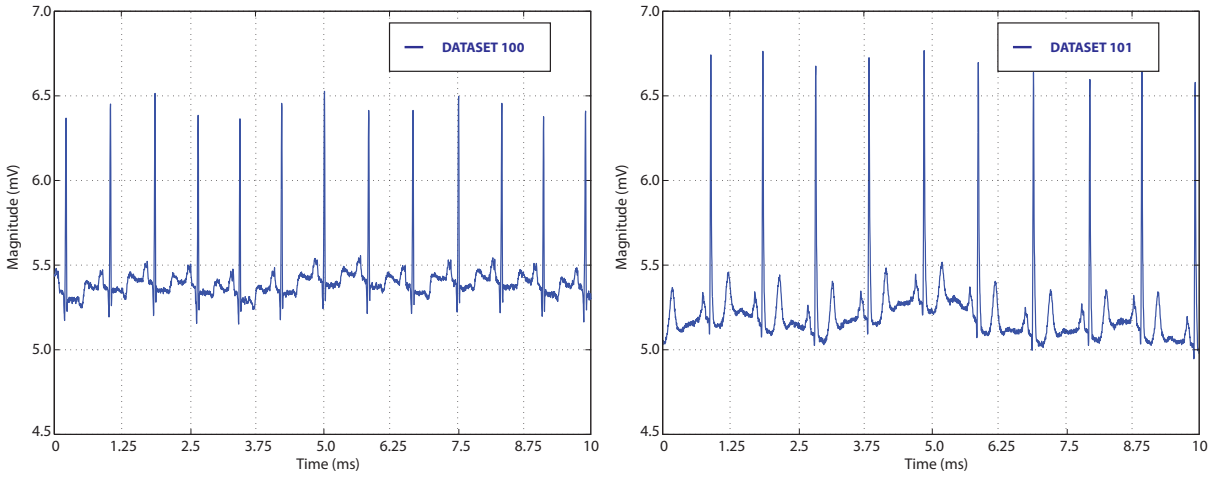
increases the mean and median RMSD errors, whereas the standard deviation errors remain almost constant.

The complete MCU time domain responses are also investigated for the dataset 100 and 101, acquired from the MIT-BIH Arrhythmia Database. Fig. 5.11 shows the "clean" sections of these datasets, whereas Fig. 5.12 presents the baseline wander, BWM1, added to these sections with its MCU-based baseline wander estimation. These estimates are then removed to obtain noise-free signal as in Fig. 5.13 and depicted with the shifted "clean" ECG signal representation for comparison. Finally, the same approach is performed on FIR filter responses of the same datasets as in Fig. 5.14.

The original datasets 100 and 101 carry residual baseline wander in addition to other noise interferences within the recording as can be seen in Fig 5.11(b). The ground truth, therefore, is not known. Even though the FIR filter response does provide a realistic estimate in baseline wander removal for the frequency content below 0.67 Hz, the higher spectral content with this approach is not filtered and during step changes, the signal might get distorted due to ringing (Gibbs phenomenon). Therefore, not all of the reported errors in Table 5.4 are "true" errors and in reality the overall system response might be better. Such an example can be seen in Fig. 5.13(a) and Fig. 5.14(a). These figures show that at the 9 second mark the FIR filter response generates an equal P and T wave magnitude. However, if the MCU response is closely investigated, it can be seen that the baseline wander response is exactly identical with the original signal generating a better estimate.

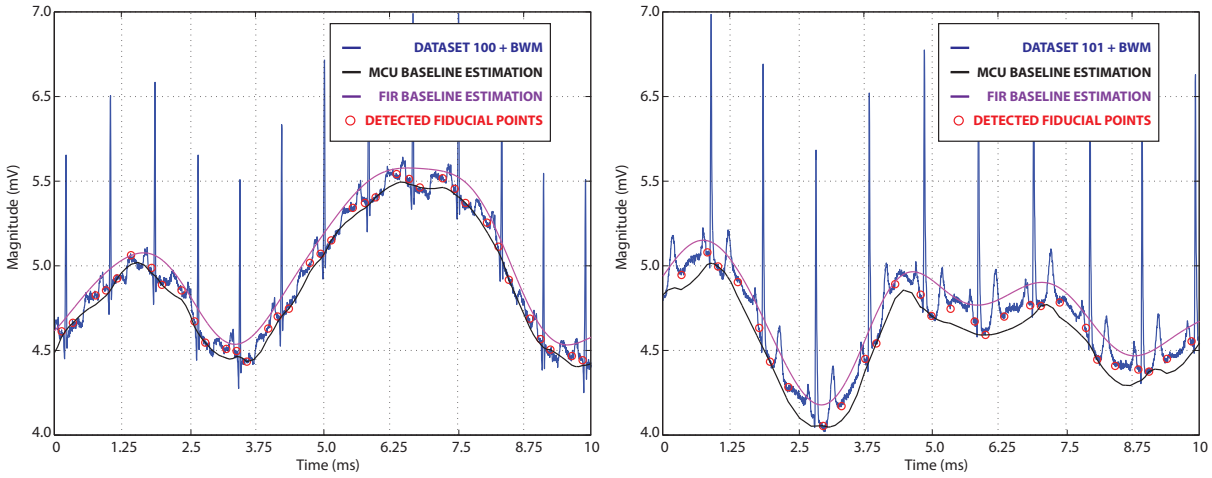
Fig. 5.13(a) and 5.13(b) show that the microcontroller-based system implementation removes the baseline wander while preserving the integrity of the ECG recording. In dataset 100, at the 2 second mark ST segment happens to have a positive slope when compared to the shifted original signal on the same plot. These shifts, however, are within the standards and are in accordance with Table 5.4 results. On the other hand, FIR filter response shows a residual offset due to the defined stop-band attenuation of the filter.

When MCU baseline estimation results are compared with the monitoring and data collection ECG devices in the market, the difference is notable. Fig. 5.15 shows the output of a recorded subject with a Shield-EKG-EMG device, which utilises Arduino Uno board for data acquisition. The Lead-II recording shows disturbance to the ECG signal integrity (ST segment and T-wave deformation) due to the high-pass filtering introduced in the analogue-front-end (AFE) design. The datasheet provides the details for the filtering stage, which utilises a single pole high-pass filter with the cut-off frequency defined at 0.16 Hz [195]. With the 1% tolerances of the utilised discrete components, and the non-linear phase distortion of high-pass filtering performed at two distinctive times - at the output of the instrumental amplifier and at the output of the regulated operational amplifier, the signal of interest is distorted. As mentioned before, a single-pole analogue filter with a cut-off frequency defined at 0.05 Hz is permitted by the standards [17–20]. In addition to the high-pass filtering, the system introduces low-pass filtering at 40 Hz and as a consequence, the ST segment is depressed, T-wave is deformed and the ECG signal loses its clinical validity.



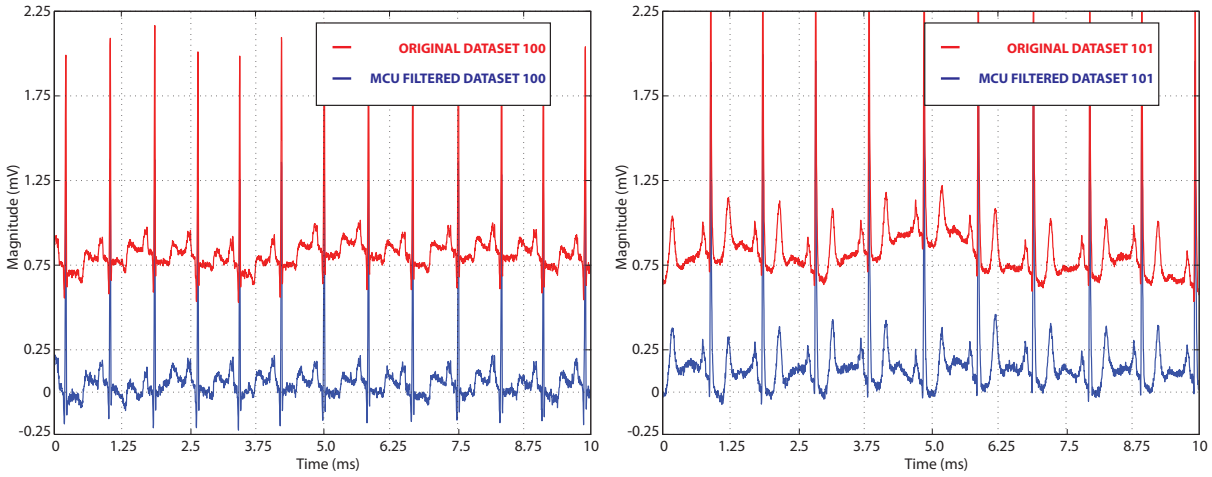
(a) "Clean" section of dataset 100 (b) "Clean" section of dataset 101

Figure 5.11.: "Clean" sections of MIT-BIH Arrhythmia Database: Datasets 100 and 101



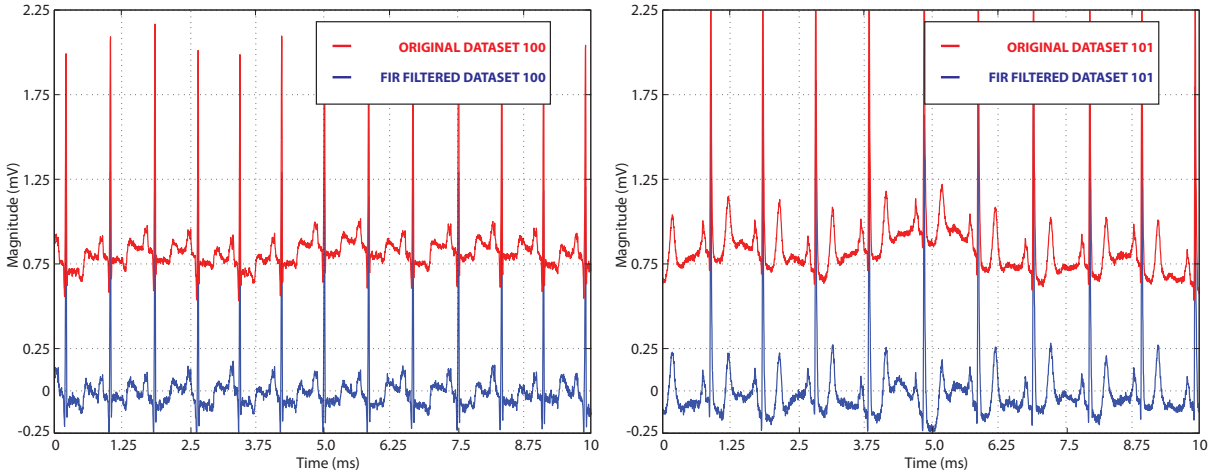
(a) MCU-based baseline estimation of dataset 100 (b) MCU-based baseline estimation of dataset 101

Figure 5.12.: MCU-based baseline estimations of "clean" sections of MIT-BIH Arrhythmia Database signals with added baseline wander from MIT-BIH Noise Stress Database



(a) MCU-based BW removal of dataset 100 (b) MCU-based BW removal of dataset 101

Figure 5.13.: MCU-based baseline wander removal of datasets 100 and 101



(a) FIR-based BW removal of dataset 101 (b) FIR-based BW removal of dataset 101

Figure 5.14.: FIR-based baseline wander removal of datasets 100 and 101

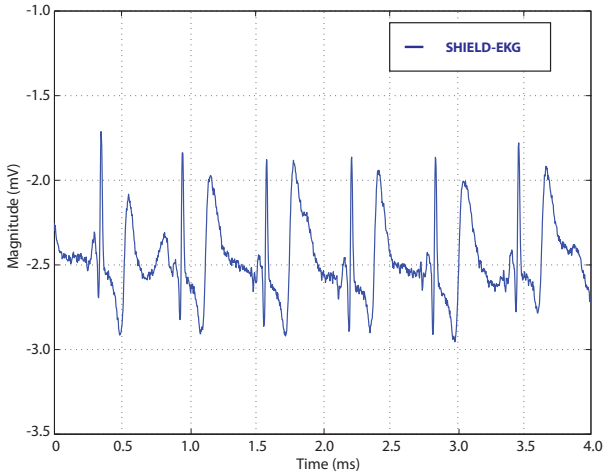


Figure 5.15.: Real recording with Shield-EKG-EMG open source hardware

5.6.2. MCU Memory & Instruction Measurements

As indicated in Section 5.4.3.1, the total memory requirement for the overall algorithm is 652 bytes. When the baseline wander algorithm is loaded on the microcontroller, total memory allocation shows 698 bytes in the RAM using the CCS memory allocation toolbox. The CCS code, however, utilises additional parameters for the UART communication and pointers for addressing variable locations.

Based on the 230400 bps baud rate and the 16 MHz MCU clock frequency, a maximum number of 40277 instructions per sample is allowed to avoid loss of information on data sampled at 360 Hz. The detailed analysis of maximum allowable number of instructions per cycles with baud rates was previously shown in Fig. 5.6. Table 5.5 shows the total number of instructions per sample generated at four distinctive sections of the ECG recording. As can be seen, the

Table 5.5.: MSP430 total number of instructions

Stage	Total number of instructions			
	Reading # 1	Reading # 2	Reading # 3	Reading # 4
Per downsampled sample				
Filter Stage (S1)	13246	12877	12942	12924
Pan & Tomp. Stage (S2)	2507	2450	2423	2506
Fiducial P.D. Stage (S3)	2070	2021	2126	2071
WPL Interp. Stage (S4)	6693	5919	4137	2911
Total	**24516**	**23267**	**21628**	**20412**
Per non-downsampled sample				
WPL Interp. Stage (S4)	6693	5919	4137	2911

table is divided into two main sections to show total number of instructions per downsampled and non-downsampled sample. As the first three stages (Filter, Pan & Tompkins, and Fiducial Point Detection) operate at the downsampled rate ($M = 3$), it follows that their total effective contribution ($\frac{instructions\ per\ cycle}{downsampling\ rate}$) to the total number of instructions per sample is on the same order as the WPL interpolation stage and the complete system is below the 40277 instructions per sample limit.

As indicated in the work of Venkat, an example of integer-float multiplication and division using C library requires 427 and 476 instructions (including type conversion from float to integer) respectively [193]. Similarly, during simulation it is observed that the filtering stage multiplication operation instruction count varies from 350 to 450 instructions. Combining this observation with the total number of computations defined in Table 5.2, it can be concluded that multiplication operations performed in each stage approximately determine the total number of instructions of that stage.

The total number of instructions, on the other hand, can be reduced with 32×32 bit hardware multiplier utilisation. Integer multiplication is evaluated in 40 instructions per cycle. However, as the MCU does not know the details about the types as these are defined by the compiler, the float representation requires additional adjustments such as multiplication of the fraction mantissas and addition of the exponents whereas fixed point number multiplication relaxes these requirements.

5.6.3. Power Consumption

As the battery-powered applications are targeted, power consumption of the embedded microprocessors is becoming more and more crucial for system design. Most silicon vendors have low-power designs, and the battery life is determined by the average current consumption, which is expressed by the function of active and low-power states of the MCU and the peripherals. Here, an approximation to the total power consumption of the MCU design is discussed and measurement results are reported.

Initially, the parameters that affect the power dissipation of the overall system design need to be quantified. As the C implementation is designed in a serial-in serial-out structure, the heart rate does not have an effect on the total power dissipation. This is because the downsampled and non-downsampled samples are processed in the same way at every clock cycle independent of the heart rate, and the power consumption is determined by the total time elapsed is in the active mode (AM), the low power mode (LPM0 [1]) and any peripherals that are being used. The duration of these instances, however, is a function of the downsampling rate and the sampling frequency in relation to the number of instructions per sample with the MCU clock frequency. The total time required by each operation is demonstrated in Fig. 5.16, where 3 ECG samples denote the downsampling rate, 1^{st} sample is the downsampled sample and the 2^{nd} and the 3^{rd} samples are the non-downsampled samples.

While processing a downsampled sample, the MCU operates in the AM longer than the non-

[1]This mode disables the CPU and MCLK whereas ACLK and SMCLK (UART clock) remain active

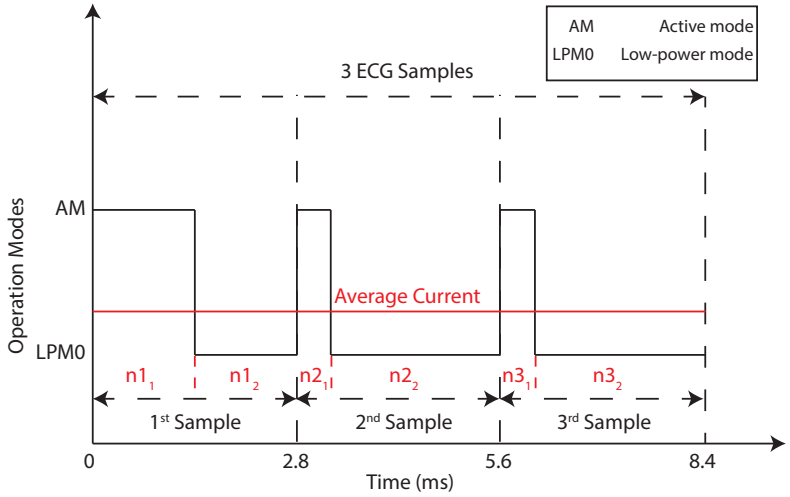


Figure 5.16.: MSP430 power dissipation modes

downsampled samples. However, after each operation the device is always put back into low-power mode to preserve current consumption before exiting the interrupt service routine (ISR) as opposed to polling. Therefore, the average current consumption based on 3 ECG samples (1 downsampled and 2 non-downsampled samples), generates the total power dissipation of the MCU. Eq. 5.6 shows the average current estimation of the overall system, where n denotes the sample number and T_{n1} and T_{n2} denote the total time for the current mode elapsed in that time frame.

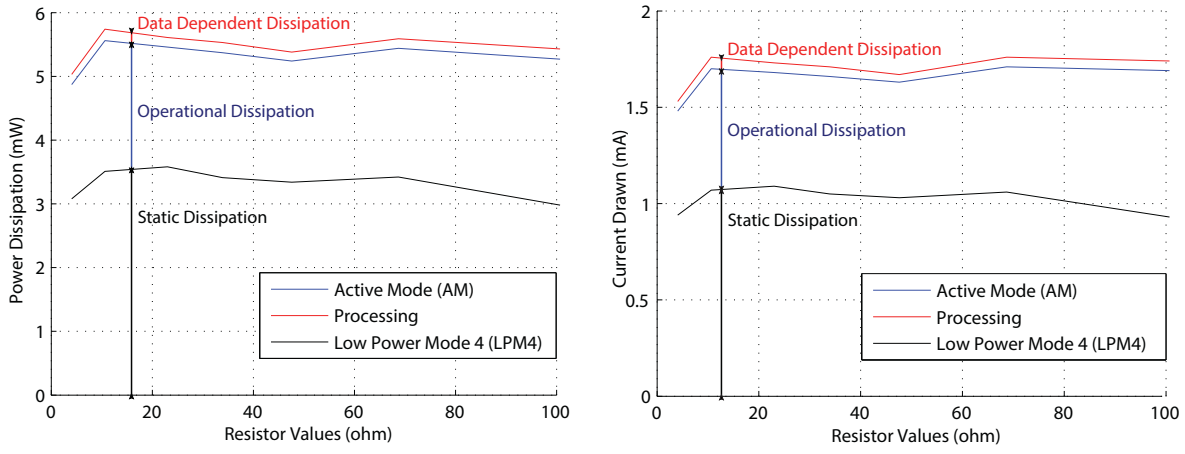
$$I_{AV} = \sum_{n=0}^{\infty} I_{AV_n}$$
$$I_{AV_n} = \frac{I_{AM_n} * T_{n1} + I_{LPM0_n} * T_{n2}}{T_{n1} + T_{n2}} \tag{5.6}$$

As covered in the previous section, the total number of instructions required for a downsampled and a non-downsampled sample is approximately equal to 25000 and 7000 (worst case conditions) instructions respectively. Based on a sampling rate of 360 Hz, each sample is required to be processed approximately in 2.78 ms. This duration is demonstrated on the x-axis of Fig. 5.16. During a downsampled sample, therefore, the MCU can be in low-power mode for approximately 1 ms, whereas this operation increases to 2.3 ms during a non-downsampled sample processing. Based on the datasheet of the MSP430, the MCU stabilises the 8 MHz MCLK in 292 ns once an interrupt is received [196]. Even with the worst conditions (6 µs) defined for such an operation, the latency at the 16 MHz clock frequency is equal to 96 cycles. When these are combined with the interrupt acceptance latency (6 cycles) and the return time from the interrupt (5 cycles), they are negligible as to the total time estimated for the low-power mode operation is larger by at least two orders of magnitude.

Based on the datasheet values with 100 % FRAM cache-hit ratio, 730 µA of current is dissipated in active mode (AM), whereas in low-power mode (LPM0) 275 µA of current is required. Using Eq. 5.6, the average theoretical current consumption on 3 samples of an ECG recording is calculated as 408 µA and the worst case pin leakage is defined as 1 µA per input. At 3.0 V supply voltage, theoretical total power dissipation of the overall system is expected to be 1.23 mW with low-power operation, whereas it is expected to increase to 2.19 mW when only the active mode is utilised.

Experimental testing of the TI MSP430FR6989 launchpad's power and current dissipation is achieved by differential measurements. Initially, the registers of the microcontroller unit (MCU) is set to low-power mode (LPM4) to disable the CPU and all the clocks. This way the static power dissipation of the launchpad is measured by recording the average current dissipated and the supply voltage of the launchpad over seven different resistor values. Following this, the MCU is set to active mode (AM) to measure the operational and the data dependent power dissipation. These measurements are recorded with the same resistor values and the average power dissipation and the current consumption of the TI MSP430FR6989 launchpad is shown in Fig. 5.17(a) and 5.17(b) respectively. As can be seen in Fig. 5.17(a), the overall power dissipation is on average 5.6 mW, whereas the data dependent power dissipation (active mode operation + processing) is measured to be on average 2.4 mW matching our theoretical calculations. The same figure also shows static and operational power dissipations where the static power dissipation is determined by the launchpad and operational power dissipation is determined by the active mode operation of the MCU without the baseline detection algorithm running.

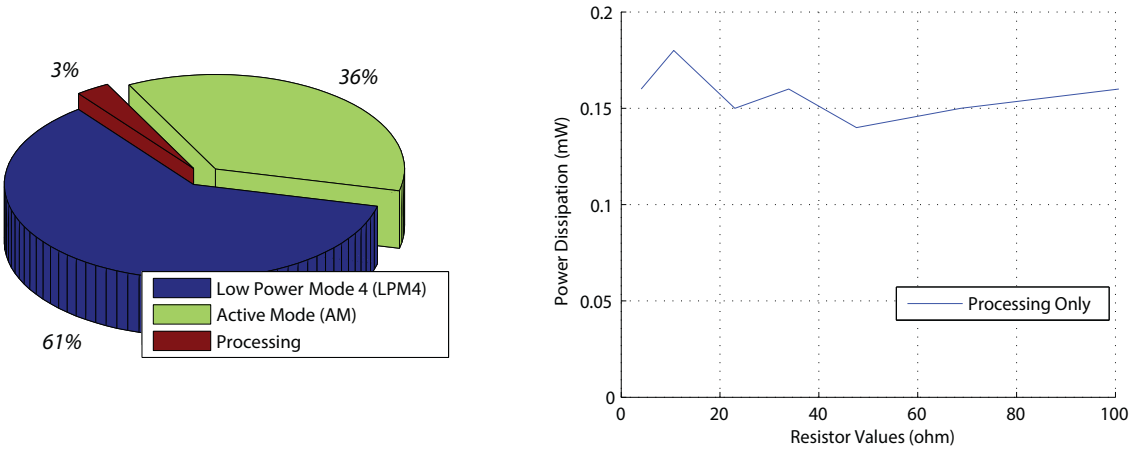
Fig. 5.18(a) shows the overall power dissipation percentages in a pie chart. As can be seen in the figure, 61% of the total power dissipation is statically dissipated, whereas the active mode operation and the data dependent operation consumes 36% and % 3% of the total power dissipation respectively. Fig. 5.18(b) depicts the data dependent power consumption in more



(a) Power dissipation

(b) Current dissipation

Figure 5.17.: TI MSP430FR6989 launchpad power and current dissipation measurements



(a) Power dissipation pie chart

(b) Processing power dissipation

Figure 5.18.: TI MSP430FR6989 launchpad additional power measurements

detail.

Based on a typical coin cell battery, CR2032, with a nominal voltage of 3.0 V - 3.4 V and a typical capacity of 225 mAh and the power dissipation measurements, 5 days of operation can be targeted respectively. With a 1000 mAh CR2477 coin cell battery, the run-time increases to 22 days of operation with the same settings respectively.

5.7. Conclusion

Due to the high computational complexity of baseline wander detection algorithms, certain limitations exist when they are applied in real-time environments. These algorithms generally rely on iterative multiplication operations which act as a bottleneck.

This chapter has demonstrated a low-power hardware realisation of the baseline wander detection and the interpolation algorithm on a MSP430FR6989 MCU. Through extensive tests on MIT-BIH Arrhythmia Database data with added baseline wander acquired from MIT-BIH Noise Stress Database, average RMSD errors of 24.2 µV mean, 21.0 µV median and 14.2 µV standard deviation are measured. These RMSD errors match with the C implementation results and the observed average RMSD differences of 2.0 µV, 1.8 µV and 0.9 µV in mean, median and standard deviation between the MCU and MATLAB tests in Chapter 3 are associated with the utilised interpolation method in baseline estimation.

The overall algorithm is arranged for the MCU to process the data in a serial-in serial-out data flow. Based on this implementation, the total number of instructions per sample is quantified and an average of four readings requires 22456 and 4915 instructions per downsampled and per non-downsampled sample respectively. Additionally, the total power dissipation is calculated as a function of the sampling rate and the total instructions per sample measurements. Based on the empirical calculations, the MCU exhibits 1.23 mW and 2.19 mW power dissipation with low-power and active mode operation respectively. Measurement results, on the other hand,

show a total power dissipation of 5.6 mW of which 2.4 mW is operational and data dependent, and 3.2 mW is static power dissipation.

When the time domain results are compared with the market devices which utilise high-pass filtering to remove the electrode offset and the baseline wander, the accuracy difference is significant. Unlike these systems, the baseline wander hardware realisation can be used to remove the baseline drift while preserving the ECG signal integrity.

Chapter 6

Conclusions

With recent advances in the medical device technology, mobile and ambulatory applications prove to be the new advancement in early detection of coronary heart diseases. Comprehensive understanding of the ECG waveform and its characteristics, therefore, has been the first priority in system-level design before addressing any noise interference present in an ECG signal. A detailed analysis provides the reader what problems are associated with each approach beforehand, and aids to address the present noise sources accurately and efficiently.

A detailed literature review shows that computationally-efficient real-time baseline wander removal techniques are yet to be proposed. As most of the reported works focus on computational algorithms, the lack of hardware realisations has led to the research presented in this thesis. Initially, a novel computationally-efficient real time baseline wander removal method is proposed. This method is then improved with the foundation of a computationally-efficient interpolation technique suitable for non-uniformly sampled signals. These algorithms are validated with the biomedical data and the MCU-based hardware implementation is tested as the final aspect of our work.

When the research objectives are re-evaluated, to what extent the original goals are achieved is one of the main questions that needs to be addressed. As initially indicated, baseline wander detection and its removal are achieved within the limits specified by the standards. While achieving this, computational complexity requirements are kept light and as the end application targeted for ambulatory designs, system evaluation and its real-time hardware implementation are demonstrated in an embedded system to achieve real-time baseline estimation.

The remainder of this chapter states the original contributions made throughout the thesis with possible future directions and is organised as follows: Section 6.1 summarises the literature review briefly; Section 6.2 lists the original contributions accreted to the literature; Section 6.3 discusses possible future directions; and Section 6.4 concludes the Thesis.

6.1. Literature Review

Initially, the interdisciplinary background makes a thorough understanding of the ECG waveform and its characteristics. The ascertained background knowledge combined with the detailed literature review in engineering simplifies the evaluation of the challenges associated with the removal of noise artefacts, in particular the baseline drift, while preserving the clinical validity of the ECG signal. Such an approach has made a break-through in this research as the conventional approaches disregarding the integrity of the ECG signal are discarded whilst investigating the literature and the reported works in detail.

As the research initiated, various studies have addressed myocardial infarction based ischaemic conditions in their motivation and utilised filters with non-linear phase characteristics that are not approved by the standards. This finding has a clear impact on this thesis in regards to the utilisation of analogue filtering. Even though the direction of the thesis is modified based on the literature findings, direct-coupled approaches with the sampling of the whole dynamic range without low-frequency high-pass filtering express the direction of the reported research.

6.2. Original Contributions

Based on the foundations of algorithms and their embedded system realisation, this thesis has made the following original contributions:

- Proposed a novel computationally-efficient baseline wander removal method based on "isoelectric" fiducial point detections. This approach preserves the ECG signal integrity with limited distortion to the ST segment when compared to the conventional algorithms and hardware solutions. The significant faults and challenges associated with baseline estimation are discussed in Chapter 2, whereas Chapter 3 describes the proposed algorithm to overcome these challenges. The key design parameters are balanced for accuracy and computationally efficiency, and both real and synthetic data test results show accurate baseline estimation with a detailed analysis when compared to most of the reported work in the literature.

- During baseline estimation, it is observed that polynomial interpolation techniques are demanding and overusing available computational resources due to quantisation noise present in the signal. Therefore, a new computationally efficient interpolation algorithm based on weighted piecewise linear interpolation has been introduced in Chapter 4. This approach finds a balance between accuracy and computational resource requirements while providing an interpolation solution for non-uniform sampled signals as required by the baseline estimation algorithm. Real and synthetic test results show accuracy improvement when compared to linear and PCHIP interpolation, whereas the algorithm proves to be implementable and less computationally demanding compared to the cubic spline approach.

- Embedded system realisation on a MCU produces real-time results for baseline wander estimation. Up-to-date, a few hardware-based solutions with limited analysis have been reported in the literature. Chapter 5 presents the practical implementation results and shows accurate baseline wander removal in detail and concludes an achievable real-time system implementation. The overall approach complies with the allowable distortion to the ST segment, and is far superior to the conventional techniques present and produces comparable results to its computational counterparts.

6.3. Future Directions

As pointed out in the previous section, this thesis has made unique contributions to the literature and there are various possible ways to pursue the presented work. In this section, possible ideas and future directions of the computational-efficient real-time baseline wander removal are discussed in more detail.

- **Confidence Levels:** As observed in real data tests, during step changes and motion artefacts the accuracy of the baseline wander estimation degrades. Therefore, an additional algorithm might introduce confidence levels and during these instances system response and ST segment distortion can be neglected.
- **Impedance Measurements:** As an extension to confidence level calculations, impedance measurements can be utilised as an additional input and the abrupt changes associated in the impedance measurements can be discarded as motion artefacts increasing the reliability of the overall system.
- **Further Power Optimisation:** Improvements on optimising the code and pursuing hardware data multiplication by utilising fixed point arithmetic can be targeted so that the total number of instructions per cycle requirements is further reduced and the total power consumption of the overall system is improved.
- **Full System Implementation:** A complete system implementation with an AFE acquired from major biomedical silicon vendors can be targeted. Within this context, the utilisation of mains as a power source in addition to battery-powered implementations need to be investigated. The effect of notch filters on low-pass filtering requirements of the baseline wander estimation algorithm might reduce the total number of instructions required. In addition, an integrated approach with instrumentation amplifiers and high resolution ADC to sample the data can be investigated to observe the system response.
- **Additional Fiducial Points:** Utilisation of additional fiducial points during T offset and P onset can be investigated. Even though such an approach increases the computational requirements, their effect on total power consumption might be negligible as the power is determined by the total number of multiplication operations, which are mostly associated with the filtering and interpolation stages.

- **Extension to Multi-Lead ECG:** Additional leads provide a better visual image of the heart's activity. Therefore, algorithms for augmented leads, Lead-I, Lead-III and the chest leads can be investigated and these might utilise the timestamp information generated on a Lead-II recording.

6.4. Concluding Remarks

It is of no doubt that wearable technology has had a great impact on our lifestyles over the last decade. We now understand the human body better and work on devices that have an effect on the lives of thousands of others.

With such a fast pace in a very short period of time, the limitations of the conventional approaches and their drawbacks are well understood and the present challenges have led us to seek for thoroughly analysed systems, which provide reliable, safe and accurate system solutions.

This research helps us comprehend the functioning of the heart thoroughly, and offer reliable and efficient solutions to improve on real-time monitoring by concentrating on computationally-efficient real-time baseline wander estimation. As not so many hardware approaches exist in the literature and most lack in-depth analysis and data validation, the work presented in this thesis provides a solution to a better understanding and leads to further advancements in ECG signal processing techniques while preserving the integrity of the ST segment.

Bibliography

[1] M. Piccolino, "Luigi galvani and animal electricity: Two centuries after the foundation of electrophysiology," *Trends in Neurosciences*, vol. 20, no. 10, pp. 443–448, 1997.

[2] J. Webster, *Medical instrumentation: Application and design.* John Wiley & Sons, 2009.

[3] T. M. Seese, H. Harasaki, G. M. Saidel, and C. R. Davies, "Characterization of tissue morphology, angiogenesis, and temperature in the adaptive response of muscle tissue to chronic heating," *Laboratory Investigation*, vol. 78, no. 12, pp. 1553–1562, 1998.

[4] E. H. Liu, G. M. Saidel, and H. Harasaki, "Model analysis of tissue responses to transient and chronic heating," *Annals of Biomedical Engineering*, vol. 31, no. 8, pp. 1007–1014, 2003.

[5] A. Alwan *et al.*, *Global status report on noncommunicable diseases 2010.* World Health Organization, 2011.

[6] World Health Organisation. (2015, January) Cardiovascular diseases. [Online]: http://www.who.int/mediacentre/factsheets/fs317/en/

[7] C. D. Mathers and D. Loncar, "Projections of global mortality and burden of disease from 2002 to 2030," *PLoS Medicine*, vol. 3, no. 11, p. e442, 2006.

[8] S. Mendis, P. Puska, B. Norrving *et al.*, *Global atlas on cardiovascular disease prevention and control.* World Health Organization, 2011.

[9] Smartheart. (2014) Smartheart. [Online]: https://www.getsmartheart.com/

[10] Sensuim. (2016) Sensuim. [Online]: http://www.sensium-healthcare.com/

[11] Fitbit. (2016) Fitbit. [Online]: https://www.fitbit.com/

[12] Apple. (2016) Apple. [Online]: http://www.apple.com/

[13] Smartheart. (2014) Smartheart. [Online]: https://www.getsmartheart.com/our-story/

[14] M. Hernandez-Silveira. (2014) Wearable technologies for clinical monitoring. [Online]: http://www.sensium-healthcare.com/

[15] J. L. Willems, C. Abreu-Lima, P. Arnaud, J. H. van Bemmel, C. Brohet, R. Degani, B. Denis, J. Gehring, I. Graham, G. van Herpen *et al.*, "The diagnostic performance of computer programs for the interpretation of electrocardiograms," *New England Journal of Medicine*, vol. 325, no. 25, pp. 1767–1773, 1991.

[16] J. Ioannidis, D. Salem, P. W. Chew, and J. Lau, "Accuracy and clinical effect of out-of-hospital electrocardiography in the diagnosis of acute cardiac ischemia: A meta-analysis," *Annals of Emergency Medicine*, vol. 37, no. 5, pp. 461–470, 2001.

[17] American National Standards Institute and Association for the Advancement of Medical Instrumentation *et al.*, *American National standard for diagnostic electrocardiographic devices (ANSI/AAMI EC11:1991/[R]2001).* Association for the Advancement of Medical Instrumentation, 1983.

[18] P. M. Okin, G. Bergman, and P. Kligfield, "Effect of ST segment measurement point on performance of standard and heart rate-adjusted ST segment criteria for the identification of coronary artery disease." *Circulation*, vol. 84, no. 1, pp. 57–66, 1991.

[19] International Electrotechnical Commission *et al.*, "Medical electrical equipment. part 2-51: Particular requirements for safety, including essential performance, of recording and analysing single channel and multichannel electrocardiographs," IEC 60601-2-51. Geneva: International Electrotechnical Commission, Tech. Rep., 2003.

[20] P. Kligfield, L. S. Gettes, J. J. Bailey, R. Childers, B. J. Deal, E. W. Hancock, G. van Herpen, J. A. Kors, P. Macfarlane, D. M. Mirvis *et al.*, "Recommendations for the standardization and interpretation of the electrocardiogram: part i: The electrocardiogram and its technology a scientific statement from the american heart association electrocardiography and arrhythmias committee, council on clinical cardiology; the american college of cardiology foundation; and the heart rhythm society endorsed by the international society for computerized electrocardiology," *Journal of the American College of Cardiology*, vol. 49, no. 10, pp. 1109–1127, 2007.

[21] R. E. Gregg, S. H. Zhou, J. M. Lindauer, E. D. Helfenbein, and K. K. Giuliano, "What is inside the electrocardiograph?" *Journal of Electrocardiology*, vol. 41, no. 1, pp. 8–14, 2008.

[22] A. Bezerianos, M. Popescu, N. Laskaris, A. Manolis, I. Hiladakis, C. Stathopoulos, and P. Cristea, "Selective noise filtering of high resolution ECG through wavelet transform," in *IEEE Proceedings of the Computers in Cardiology.* IEEE, 1996, pp. 637–640.

[23] E. J. Berbari, "High-resolution electrocardiography." *Critical Reviews in Biomedical Engineering*, vol. 16, no. 1, pp. 67–103, 1987.

[24] S. Narayanaswamy, "High resolution electrocardiography," *Indian Pacing and Electrophysiology Journal*, vol. 2, no. 2, p. 50, 2002.

[25] W. Kühnel, *Color atlas of cytology, histology, and microscopic anatomy.* Stuttgart; New York: Thieme, 2003.

[26] E. B. Heather Ketchum. (2015, June) OU human physiology: Cardiac muscle and electrical activity. [Online]: http://cnx.org/contents/9f1fbbf2-055f-45db-a336-aa37a485cd5d@1

[27] S. K. Pat F. Bass. (2016, May) Anatomy and function of the electrical system. [Online]: https://www.urmc.rochester.edu/Encyclopedia/Content.aspx?ContentTypeID=90&ContentID=P01762

[28] A. Houghton and D. Gray, *Making Sense of the ECG: A hands-on guide.* Hachette UK, 2008.

[29] L. R. Piero. (2015, August) Electrocardiografia basica. [Online]: https://www.amperordirect.com/pc/help-ecg-monitor/z-what-is-ecg.html

[30] Pixcove. (2015) Pixcove. [Online]: http://www.pixcove.com/ribs-beams-medical-minimum-health-human-humanoid-skeleton-health-bone-jawbone-skeletal-anatomy-medical-x-ray-structure-chest/

[31] D. B. Foster, "Myocardial infarction," *Twelve-Lead Electrocardiography: Theory and Interpretation*, pp. 64–83, 2007.

[32] K. Thygesen, J. S. Alpert, A. S. Jaffe, H. D. White, M. L. Simoons, B. R. Chaitman, H. A. Katus, F. S. Apple, B. Lindahl, D. A. Morrow *et al.*, "Third universal definition of myocardial infarction," *Journal of the American College of Cardiology*, vol. 60, no. 16, pp. 1581–1598, 2012.

[33] P. G. Steg, S. K. James, D. Atar, L. P. Badano, C. B. Lundqvist, M. A. Borger, C. Di Mario, K. Dickstein, G. Ducrocq, F. Fernandez-Aviles *et al.*, "ESC guidelines for the management of acute myocardial infarction in patients presenting with ST-segment elevation," *European Heart Journal*, p. ehs215, 2012.

[34] E. A. Amsterdam, N. K. Wenger, R. G. Brindis, D. E. Casey, T. G. Ganiats, D. R. Holmes, A. S. Jaffe, H. Jneid, R. F. Kelly, M. C. Kontos *et al.*, "2014 AHA/ACC guideline for the management of patients with non–ST-elevation acute coronary syndromes: A report of the american college of cardiology/american heart association task force on practice guidelines," *Journal of the American College of Cardiology*, vol. 64, no. 24, pp. e139–e228, 2014.

[35] P. T. O'Gara, F. G. Kushner, D. D. Ascheim, D. E. Casey, M. K. Chung, J. A. De Lemos, S. M. Ettinger, J. C. Fang, F. M. Fesmire, B. A. Franklin *et al.*, "2013 ACCF/AHA guideline for the management of ST-elevation myocardial infarction: A report of the american college of cardiology foundation/american heart association task force on practice guidelines," *Journal of the American College of Cardiology*, vol. 61, no. 4, pp. e78–e140, 2013.

[36] E. R. Laskowski. (2015, August) What's a normal resting heart rate? [Online]: http://www.mayoclinic.org/healthy-lifestyle/fitness/expert-answers/heart-rate/faq-20057979

[37] American Heart Association. (2014, October) Bradycardia. [Online]: http://www.heart.org/HEARTORG/Conditions/Arrhythmia/AboutArrhythmia/Bradycardia-Slow-Heart-Rate_UCM_302016_Article.jsp#.Vy4DlRUrJTY

[38] J. J. Bailey, A. S. Berson, A. Garson Jr, L. G. Horan, P. W. Macfarlane, D. W. Mortara, and C. Zywietz, "Recommendations for standardization and specifications in automated electrocardiography: Bandwidth and digital signal processing. a report for health professionals by an ad hoc writing group of the committee on electrocardiography and cardiac electrophysiology of the council on clinical cardiology, american heart association." *Circulation*, vol. 81, no. 2, p. 730, 1990.

[39] J. García-Niebla, P. Llontop-García, J. I. Valle-Racero, G. Serra-Autonell, V. N. Batchvarov, and A. B. De Luna, "Technical mistakes during the acquisition of the electrocardiogram," *Annals of Noninvasive Electrocardiology*, vol. 14, no. 4, pp. 389–403, 2009.

[40] G. M. Friesen, T. C. Jannett, M. A. Jadallah, S. L. Yates, S. R. Quint, and H. T. Nagle, "A comparison of the noise sensitivity of nine QRS detection algorithms," *IEEE Transactions on Biomedical Engineering*, vol. 37, no. 1, pp. 85–98, 1990.

[41] B. M. Nigg and W. Herzog, *Biomechanics of the musculo-skeletal system.* John Wiley & Sons, 2007.

[42] M. Yelderman, B. Widrow, J. M. Cioffi, E. Hesler, and J. A. Leddy, "ECG enhancement by adaptive cancellation of electrosurgical interference," *IEEE Transactions on Biomedical Engineering*, no. 7, pp. 392–398, 1983.

[43] W. R. Bennett, "Spectra of quantized signals," *Bell System Technical Journal*, vol. 27, no. 3, pp. 446–472, 1948.

[44] A. S. Berson and H. V. Pipberger, "Electrocardiographic distortions caused by inadequate high-frequency response of direct-writing electrocardiographs," *American Heart Journal*, vol. 74, no. 2, pp. 208–218, 1967.

[45] C. Zywietz, G. Wagner, B. Scherlag *et al.*, "Sampling rate of ECGs in relation to measurement accuracy," *Computerized Interpretation of the Electrocardiogram. New York, NY: Engineering Foundation*, pp. 122–5, 1986.

[46] Association for the Advancement of Medical Instrumentation, "ANSI/AAMI EC11: Diagnostic electrocardiographic devices." 1991-2001.

[47] F. W. Breyfogle III, *Implementing six sigma: Smarter solutions using statistical methods.* John Wiley & Sons, 2003.

[48] Texas Instruments. (2011, May) Thermal noise analysis in ECG applications. [Online]: http://www.ti.com/lit/an/sbaa185/sbaa185.pdf

[49] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, N.-C. Yen, C. C. Tung, and H. H. Liu, "The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis," *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 454, no. 1971, pp. 903–995, 1998.

[50] M. Blanco-Velasco, B. Weng, and K. Barner, "ECG signal denoising and baseline wander correction based on the empirical mode decomposition," *Computers in Biology and Medicine*, vol. 38, no. 1, pp. 1–13, 2008.

[51] B. Weng, M. Blanco-Velasco, and K. E. Barner, "Baseline wander correction in ECG by the empirical mode decomposition," in *IEEE Proceedings of the Bioengineering Conference*. IEEE, 2006, pp. 135–136.

[52] M. Blanco-Velasco, B. Weng, and K. Barner, "A new ECG enhancement algorithm for stress ECG tests," in *IEEE Proceedings of the Computers in Cardiology*. IEEE, 2006, pp. 917–920.

[53] K.-M. Chang, "Arrhythmia ECG noise reduction by ensemble empirical mode decomposition," *Sensors*, vol. 10, no. 6, pp. 6063–6080, 2010.

[54] J. Jenitta and A. Rajeswari, "Denoising of ECG signal based on improved adaptive filter with EMD and EEMD," in *IEEE Proceedings of the Information & Communication Technologies*. IEEE, 2013, pp. 957–962.

[55] G. Singh, G. Kaur, and V. Kumar, "ECG denoising using adaptive selection of IMFs through EMD and EEMD," in *IEEE Proceedings of the Data Science & Engineering*. IEEE, 2014, pp. 228–231.

[56] S. Anapagamini and R. Rajavel, "Removal of artifacts in ECG using empirical mode decomposition," in *IEEE Proceedings of the Communications and Signal Processing*. IEEE, 2013, pp. 288–292.

[57] N. Pan, V. Mang, M. P. Un *et al.*, "Accurate removal of baseline wander in ECG using empirical mode decomposition," in *IEEE Proceedings of the Noninvasive Functional Source Imaging of the Brain and Heart and the International Conference on Functional Biomedical Imaging*. IEEE, 2007, pp. 177–180.

[58] Z. Zhidong and M. Chan, "A novel cancellation method of powerline interference in ECG signal based on EMD and adaptive filter," in *IEEE Proceedings of the Communication Technology*. IEEE, 2008, pp. 517–520.

[59] Z. Zhidong and L. Juan, "Baseline wander removal of ECG signals using empirical mode decomposition and adaptive filter," in *IEEE Proceedings of the Bioinformatics and Biomedical Engineering*. IEEE, 2010, pp. 1–3.

[60] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent component analysis*. John Wiley & Sons, 2004, vol. 46.

[61] T.-P. Jung, S. Makeig, T.-W. Lee, M. J. McKeown, G. Brown, A. J. Bell, and T. J. Sejnowski, "Independent component analysis of biomedical signals," in *Proceedings of the International Workshop on Independent Component Analysis and Signal Separation*, 2000, pp. 633–644.

[62] C. J. James and C. W. Hesse, "Independent component analysis for biomedical signals," *Physiological measurement*, vol. 26, no. 1, p. R15, 2004.

[63] M. Chawla, "PCA and ICA processing methods for removal of artifacts and noise in electrocardiograms: A survey and comparison," *Applied Soft Computing*, vol. 11, no. 2, pp. 2216–2226, 2011.

[64] T. He, G. Clifford, and L. Tarassenko, "Application of independent component analysis in removing artefacts from the electrocardiogram," *Neural Computing & Applications*, vol. 15, no. 2, pp. 105–116, 2006.

[65] N. Jain and D. K. Shakya, "Denoising baseline wander noise from electrocardiogram signal using fast ICA with multiple adjustments," *International Journal of Computer Applications*, vol. 99, no. 2, pp. 34–39, 2014.

[66] M. Sarfraz, F. Li, and M. Javed, "A comparative study of ICA algorithms for ECG signal processing," in *Proceedings of the International Conference on Advances in Computing and Artificial Intelligence*. ACM, 2011, pp. 135–138.

[67] J. M. Tanskanen and J. J. Viik, *Independent Component Analysis in ECG Signal Processing*. INTECH Open Access Publisher, 2012.

[68] Z. Barati and A. Ayatollahi, "Baseline wandering removal by using independent component analysis to single-channel ECG data," in *IEEE Proceedings of the Biomedical and Pharmaceutical Engineering*. IEEE, 2006, pp. 152–156.

[69] J. O. Wisbeck, A. K. Barros, A. K. B. Yy, and R. G. Ojeda, "Application of ICA in the separation of breathing artifacts in ECG signals," 1998.

[70] P. Mishra and S. K. Singla, "Artifact removal from biosignal using fixed point ICA algorithm for pre-processing in biometric recognition," *Measurement Science Review*, vol. 13, no. 1, pp. 7–11, 2013.

[71] A. Rashid, I. M. Qureshi, A. Saleem *et al.*, "Electrocardiogram signal processing for baseline noise removal using blind source separation techniques: A comparative analysis,"

in *IEEE Proceedings of the Machine Learning and Cybernetics*, vol. 4. IEEE, 2011, pp. 1756–1761.

[72] M. W. Rahman and M. A. Riheen, "Compatibility of mother wavelet functions with the electrocardiographic signal," in *IEEE Proceedings of the Informatics, Electronics & Vision*. IEEE, 2013, pp. 1–4.

[73] R. Von Borries, J. Pierluissi, and H. Nazeran, "Wavelet transform-based ECG baseline drift removal for body surface potential mapping," in *IEEE Proceedings of the Engineering in Medicine and Biology Society*. IEEE, 2006, pp. 3891–3894.

[74] K. Park, K. Lee, and H. Yoon, "Application of a wavelet adaptive filter to minimise distortion of the ST-segment," *Medical and Biological Engineering and Computing*, vol. 36, no. 5, pp. 581–586, 1998.

[75] X. Li, T. Wang, P. Zhou, and H. Feng, "ST-T complex automatic analysis of the electrocardiogram signals based on wavelet transform," in *IEEE Proceedings of the Bioengineering Conference*. IEEE, 2003, pp. 144–145.

[76] W. Hao, Y. Chen, and Y. Xin, "ECG baseline wander correction by mean-median filter and discrete wavelet transform," in *IEEE Proceedings of the Engineering in Medicine and Biology Society, EMBC*. IEEE, 2011, pp. 2712–2715.

[77] O. Sayadi and M. B. Shamsollahi, "Multiadaptive bionic wavelet transform: Application to ECG denoising and baseline wandering reduction," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, no. 1, pp. 1–11, 2007.

[78] S. K. Yadav, R. Sinha, and P. K. Bora, "Electrocardiogram signal denoising using non-local wavelet transform domain filtering," *Signal Processing, IET*, vol. 9, no. 1, pp. 88–96, 2015.

[79] K. Daqrouq, "ECG baseline wandering reduction using discrete wavelet transform," *Asian Journal of Information Technology*, vol. 4, no. 11, pp. 989–995, 2005.

[80] A. Sargolzaei, K. Faez, and S. Sargolzaei, "A new robust wavelet based algorithm for baseline wandering cancellation in ECG signals," in *IEEE Proceedings of the Signal and Image Processing Applications*. IEEE, 2009, pp. 33–38.

[81] D. Zhang, "Wavelet approach for ECG baseline wander correction and noise reduction," in *IEEE Proceedings of the Engineering in Medicine and Biology Society*. IEEE, 2005, pp. 1212–1215.

[82] B. Arvinti, D. Toader, M. Costache, and A. Isar, "Electrocardiogram baseline wander removal using stationary wavelet approximations," in *IEEE Proceedings of the Optimization of Electrical and Electronic Equipment*. IEEE, 2010, pp. 890–895.

[83] R. Kumari, S. Bharathi, and V. Sadasivam, "Design of optimal discrete wavelet for ECG signal using orthogonal filter bank," in *IEEE Proceedings of the Conference on Computational Intelligence and Multimedia Applications*, vol. 1. IEEE, 2007, pp. 525–529.

[84] B. B. Mandelbrot and J. W. Van Ness, "Fractional brownian motions, fractional noises and applications," *SIAM Review*, vol. 10, no. 4, pp. 422–437, 1968.

[85] Y. Yamamoto and R. L. Hughson, "Coarse-graining spectral analysis: New method for studying heart rate variability," *Journal of Applied Physiology*, vol. 71, no. 3, pp. 1143–1150, 1991.

[86] S. Agrawal and A. Gupta, "Removal of baseline wander in ECG using the statistical properties of fractional brownian motion," in *IEEE Proceedings of the Electronics, Computing and Communication Technologies*. IEEE, 2013, pp. 1–6.

[87] S. Agrawal and A. Gupta, "Fractal and EMD based removal of baseline wander and powerline interference from ECG signals," *Computers in Biology and Medicine*, vol. 43, no. 11, pp. 1889–1899, 2013.

[88] C. Meyer and H. Keiser, "Electrocardiogram baseline noise estimation and removal using cubic splines and state-space computation techniques," *Computers and Biomedical Research*, vol. 10, no. 5, pp. 459–470, 1977.

[89] F. Badilini, A. Moss, and E. Titlebaum, "Cubic spline baseline estimation in ambulatory ECG recordings for the measurement of ST segment displacements," in *IEEE Proceedings of the Engineering in Medicine and Biology Society*, vol. 13. IEEE, 1991, pp. 584–585.

[90] J. N. Froning, M. D. Olson, and V. F. Froelicher, "Problems and limitations of ECG baseline estimation and removal using a cubic spline technique during exercise ECG testing: Recommendations for proper implementation," *Journal of Electrocardiology*, vol. 21, pp. S149–S157, 1988.

[91] J. R. Gradwohl, E. W. Pottala, M. R. Horton, and J. J. Bailey, "Comparison of two methods for removing baseline wander in the ECG," in *IEEE Proceedings of the Computers in Cardiology*. IEEE, 1988, pp. 493–496.

[92] L. F. Brown and S. P. Arunachalam, "Real-time TP knot algorithm for baseline wander noise removal from the electrocardiogram," in *IEEE Proceedings of the Rocky Mountain Bioengineering Symposium*. IEEE, 2009.

[93] A. Fasano and V. Villani, "Baseline wander removal for bioelectrical signals by quadratic variation reduction," *Signal Processing*, vol. 99, pp. 48–57, 2014.

[94] A. Fasano and V. Villani, "ECG baseline wander removal and impact on beat morphology: A comparative analysis," in *IEEE Proceedings of the Computing in Cardiology*. IEEE, 2013, pp. 1167–1170.

[95] A. Fasano, V. Villani, and L. Vollero, "Baseline wander estimation and removal by quadratic variation reduction," in *IEEE Proceedings of the Engineering in Medicine and Biology Society*. IEEE, 2011, pp. 977–980.

[96] A. Fasano, V. Villani, and L. Vollero, "Fast ECG baseline wander removal preserving the ST segment," in *Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies*. ACM, 2011, p. 56.

[97] L. She, Z. Xu, S. Zhang, and Y. Song, "De-noisng of ECG based on EMD improved-thresholding and mathematical morphology operation," in *IEEE Proceedings of the Biomedical Engineering and Informatics*, vol. 2. IEEE, pp. 838–842.

[98] S. Taouli and F. Bereksi-Reguig, "Noise and baseline wandering suppression of ECG signals by morphological filter," *Journal of Medical Engineering & Technology*, vol. 34, no. 2, pp. 87–96, 2010.

[99] T. Ji, Z. Lu, Q. Wu, and Z. Ji, "Baseline normalisation of ECG signals using empirical mode decomposition and mathematical morphology," *Electronics Letters*, vol. 44, no. 2, p. 1, 2008.

[100] C.-H. Chu and E. J. Delp, "Impulsive noise suppression and background normalization of electrocardiogram signals using morphological operators." *IEEE Transactions on Biomedical Engineering*, vol. 36, no. 2, pp. 262–273, 1989.

[101] Y. Sun, K. L. Chan, and S. M. Krishnan, "ECG signal conditioning by morphological filtering," *Computers in Biology and Medicine*, vol. 32, no. 6, pp. 465–479, 2002.

[102] C.-I. Ieong, M.-I. Vai, P.-U. Mak, and P.-I. Mak, "ECG heart beat detection via mathematical morphology and quadratic spline wavelet transform," *IEEE Proceedings of the International Conference on Consumer Electronics*, pp. 609–610, 2011.

[103] S. H. Oğuz and M. H. Asyali, "A morphology based algorithm for baseline wander elimination in ECG records," in *IEEE Proceedings of the Biomedical Engineering Days*. IEEE, 1992, pp. 157–160.

[104] M. Dai and S.-L. Lian, "Removal of baseline wander from dynamic electrocardiogram signals," in *IEEE Proceedings of the Image and Signal Processing*. IEEE, 2009, pp. 1–4.

[105] J. M. Leski and N. Henzel, "ECG baseline wander and powerline interference reduction using nonlinear filter bank," *Signal Processing*, vol. 85, no. 4, pp. 781–793, 2005.

[106] E. W. Pottala, J. J. Bailey, M. R. Horton, and J. R. Gradwohl, "Suppression of baseline wander in the ECG using a bilinearly transformed, null-phase filter," *Journal of Electrocardiology*, vol. 22, pp. 243–247, 1990.

[107] R. A. Frankel, E. W. Pottala, R. W. Bowser, and J. J. Bailey, "A filter to suppress ECG baseline wander and preserve st-segment accuracy in a real-time environment," *Journal of Electrocardiology*, vol. 24, no. 4, pp. 315–323, 1991.

[108] V. Shusterman, S. I. Shah, A. Beigel, and K. P. Anderson, "Enhancing the precision of ECG baseline correction: Selective filtering and removal of residual error," *Computers and Biomedical Research*, vol. 33, no. 2, pp. 144–160, 2000.

[109] J. Van Alste and T. Schilder, "Removal of base-line wander and power-line interference from the ECG by an efficient FIR filter with a reduced number of taps," *IEEE Transactions on Biomedical Engineering*, no. 12, pp. 1052–1060, 1985.

[110] J. Van Alste, W. Van Eck, and O. Herrmann, "ECG baseline wander reduction using linear phase filters," *Computers and Biomedical Research*, vol. 19, no. 5, pp. 417–427, 1986.

[111] V. de Pinto, "Filters for the reduction of baseline wander and muscle artifact in the ECG," *Journal of Electrocardiology*, vol. 25, pp. 40–48, 1992.

[112] K. S. Kumar, B. Yazdanpanah, and P. R. Kumar, "Removal of noise from electrocardiogram using digital FIR and IIR filters with various methods," in *IEEE Proceedings of the Communications and Signal Processing*. IEEE, 2015, pp. 0157–0162.

[113] K. K. Patro and P. R. Kumar, "De-noising of ECG raw signal by cascaded window based digital filters configuration," in *IEEE Proceedings of the Power, Communication and Information Technology Conference*. IEEE, 2015, pp. 120–124.

[114] T. Singh, P. Agarwal, and V. Pandey, "ECG baseline noise removal techniques using window based fir filters," in *IEEE Proceedings of the Medical Imaging, m-Health and Emerging Communication Systems*. IEEE, 2014, pp. 131–136.

[115] R. Warlar and C. Eswaran, "Integer coefficient bandpass filter for the simultaneous removal of baseline wander, 50 and 100 hz interference from the ECG," *Medical and Biological Engineering and Computing*, vol. 29, no. 3, pp. 333–336, 1991.

[116] D. Jingwei and J. Wenwen, "Design of digital filter on ECG signal processing," in *IEEE Proceedings of the Instrumentation and Measurement, Computer, Communication and Control*. IEEE, 2015, pp. 1272–1275.

[117] N. V. Thakor and Y.-S. Zhu, "Applications of adaptive filtering to ECG analysis: Noise cancellation and arrhythmia detection," *IEEE Transactions on Biomedical Engineering*, vol. 38, no. 8, pp. 785–794, 1991.

[118] M. Thodetil and S. L. Gutta, "Analysis of removing noise from modeled ECG signals by using adaptive noise cancellation model," *Analysis*, vol. 4, no. 12, 2015.

[119] L. Shetty, "Electrocardiogram preprocessing using weiner filter & least mean square algorithm."

[120] M. Z. U. Rahman, R. A. Shaik, and D. R. K. Reddy, "Noise cancellation in ECG signals using computationally simplified adaptive filtering techniques: Application to biotelemetry," *Signal Processing: An International Journal*, vol. 3, no. 5, pp. 1–12, 2009.

[121] B. Paul and P. Mythili, "ECG noise removal using GA tuned sign-data least mean square algorithm," in *IEEE Proceedings of the Advanced Communication Control and Computing Technologies*. IEEE, 2012, pp. 100–103.

[122] M. Z. U. Rahman, R. A. Shaik, and D. Reddy, "Baseline wander and power line interference elimination from cardiac signals using error nonlinearity LMS algorithm," in *IEEE Proceedings of the Systems in Medicine and Biology*. IEEE, 2010, pp. 217–220.

[123] U. Rahman, M. Zia, R. A. Shaik, and D. Reddy, "Adaptive noise removal in the ECG using the block LMS algorithm," in *IEEE Proceedings of the Adaptive Science & Technology*. IEEE, 2009, pp. 380–383.

[124] T. Gowri, P. R. Kumar, D. K. Reddy, and U. Z. Rahman, "Removal of artifacts from electrocardiogram using efficient dead zone leaky LMS adaptive algorithm," in *Proceedings of the Advanced Computing, Networking and Informatics*. Springer, 2016, pp. 619–630.

[125] R. Jané, P. Laguna, N. V. Thakor, and P. Caminal, "Adaptive baseline wander removal in the ECG: Comparative analysis with cubic spline technique," in *IEEE Proceedings of the Computers in Cardiology*. IEEE, 1992, pp. 143–146.

[126] P. Laguna, R. Jané, and P. Camina, "Adaptive filtering of ECG baseline wander," in *IEEE Proceedings of the Engineering in Medicine and Biology Society*, vol. 2. IEEE, 1992, pp. 508–509.

[127] A. C. Mugdha, F. S. Rawnaque, and M. U. Ahmed, "A study of recursive least squares (RLS) adaptive filter algorithm in noise removal from ECG signals," in *IEEE Proceedings of the Informatics, Electronics & Vision*. IEEE, 2015, pp. 1–6.

[128] Y. Wu and R. M. Rangayyan, "An algorithm for evaluating the performance of adaptive filters for the removal of artifacts in ECG signals," in *IEEE Proceedings of the Electrical and Computer Engineering*. IEEE, 2007, pp. 864–867.

[129] I. Romero, D. Geng, and T. Berset, "Adaptive filtering in ECG denoising: A comparative study," in *IEEE Proceedings of the Computing in Cardiology*. IEEE, 2012, pp. 45–48.

[130] C. Chandrakar and M. Kowar, "Denoising ECG signals using adaptive filter algorithm," *International Journal of Soft Computing and Engineering*, vol. 2, no. 1, pp. 120–123, 2012.

[131] M. Mneimneh, E. Yaz, M. Johnson, and R. Povinelli, "An adaptive kalman filter for removing baseline wandering in ECG signals," in *IEEE Proceedings of the Computers in Cardiology*. IEEE, 2006, pp. 253–256.

[132] R. Vullings, B. De Vries, and J. W. Bergmans, "An adaptive kalman filter for ECG signal enhancement," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 4, pp. 1094–1103, 2011.

[133] O. Sayadi and M. B. Shamsollahi, "ECG denoising and compression using a modified extended kalman filter structure," *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 9, pp. 2240–2248, 2008.

[134] M.-H. Lee, K.-K. Shyu, P.-L. Lee, C.-M. Huang, and Y.-J. Chiu, "Hardware implementation of EMD using DSP and FPGA for online signal processing," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 6, pp. 2473–2481, 2011.

[135] L. Wang, M. I. Vai, P. U. Mak, and C. I. Ieong, "Hardware-accelerated implementation of EMD," in *IEEE Proceedings of the Biomedical Engineering and Informatics*, vol. 2. IEEE, 2010, pp. 912–915.

[136] S. Kim, R. Yazicioglu, T. Torfs, B. Dilpreet, P. Julien, and C. Van Hoof, "A 2.4μa continuous-time electrode-skin impedance measurement circuit for motion artifact monitoring in ECG acquisition systems," in *IEEE Proceedings of the VLSI Circuits*. IEEE, 2010, pp. 219–220.

[137] N. Van Helleputte, S. Kim, H. Kim, J. P. Kim, C. Van Hoof, and R. Yazicioglu, "A 160 a biopotential acquisition ASIC with fully integrated IA and motion-artifact suppression," in *IEEE Proceedings of the Solid-State Circuits Conference Digest of Technical Papers*, feb. 2012, pp. 118 –120.

[138] Q. Fan, F. Sebastiano, J. Huijsing, and K. Makinwa, "A 1.8mu w 60 nv /$surd$ hz capacitively-coupled chopper instrumentation amplifier in 65 nm CMOS for wireless sensor nodes," *IEEE Journal of Solid-State Circuits*, no. 99, pp. 1–1, 2011.

[139] R. Yazicioglu, T. Torfs, P. Merken, J. Penders, V. Leonov, R. Puers, B. Gyselinckx, and C. Van Hoof, "Ultra-low-power biopotential interfaces and their applications in wearable and implantable systems," *Microelectronics Journal*, vol. 40, no. 9, pp. 1313–1321, 2009.

[140] J. Xu, R. Yazicioglu, B. Grundlehner, P. Harpe, K. Makinwa, and C. Van Hoof, "A 8-channel active electrode system for EEG monitoring," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 5, no. 6, pp. 555–567, 2011.

[141] R. Yazicioglu, P. Merken, R. Puers, and C. Van Hoof, "A 60mu w 60 nv/$surd$ hz readout front-end for portable biopotential acquisition systems," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 5, pp. 1100–1110, 2007.

[142] T. Niederhauser, T. Marisa, L. Kohler, A. Haeberlin, R. A. Wildhaber, R. Abächerli, J. Goette, M. Jacomet, and R. Vogel, "A baseline wander tracking system for artifact rejection in long-term electrocardiography," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 10, no. 1, pp. 255–265, 2016.

[143] Y. Lian and P. Ho, "ECG noise reduction using multiplier-free FIR digital filters," in *IEEE Proceedings of the Signal Processing*, vol. 3. IEEE, 2004, pp. 2198–2201.

[144] Alivecor. (2016) Alivecor. [Online]: http://www.alivecor.com/

[145] L. A. Lipsitz, J. Mietus, G. B. Moody, and A. L. Goldberger, "Spectral characteristics of heart rate variability before and during postural tilt. relations to aging and risk of syncope." *Circulation*, vol. 81, no. 6, pp. 1803–1810, 1990.

[146] B. Boashash, *Time-frequency signal analysis and processing: A comprehensive reference*. Academic Press, 2015.

[147] V. X. Afonso, "ECG QRS detection," *Biomedical Digital Signal Processing*, pp. 237–264, 1993.

[148] J. Pan and W. J. Tompkins, "A real-time QRS detection algorithm," *IEEE Transactions on Biomedical Engineering*, no. 3, pp. 230–236, 1985.

[149] P. S. Hamilton and W. J. Tompkins, "Quantitative investigation of QRS detection rules using the MIT/BIH arrhythmia database," *IEEE Transactions on Biomedical Engineering*, no. 12, pp. 1157–1165, 1986.

[150] A. Ligtenberg and M. Kunt, "A robust-digital QRS-detection algorithm for arrhythmia monitoring," *Computers and Biomedical Research*, vol. 16, no. 3, pp. 273–286, 1983.

[151] R. Karthik, "ECG simulation using MATLAB," *B.Eng. dissertation, Anna University, Chennai, India*, 2007.

[152] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, 2000.

[153] I. Silva and G. Moody, "An open-source toolbox for analysing and processing PhysioNet databases in MATLAB and Octave," *Journal of Open Research Software*, vol. 2, no. 1, p. e27, 2014.

[154] R. W. Schafer and L. R. Rabiner, "A digital signal processing approach to interpolation," *Proceedings of the IEEE*, vol. 61, no. 6, pp. 692–702, 1973.

[155] M. Elgendi, M. Jonkman, and F. DeBoer, "Frequncy bands effects on QRS detection," *Pan*, vol. 5, p. 15Hz, 2010.

[156] O. Guven, A. Eftekhar, R. Hoshyar, G. Frattini, W. Kindt, and T. G. Constandinou, "Realtime ECG baseline removal: An isoelectric point estimation approach," in *IEEE Proceedings of the Biomedical Circuits and Systems Conference*. IEEE, 2014, pp. 29–32.

[157] S. W. Smith *et al.*, "The scientist and engineer's guide to digital signal processing," 1997.

[158] B. P. Lathi and R. A. Green, *Essentials of digital signal processing*. Cambridge University Press, 2014.

[159] A. V. Oppenheim, R. W. Schafer, J. R. Buck *et al.*, *Discrete-time signal processing*. Prentice hall Englewood Cliffs, NJ, 1989, vol. 2.

[160] F. A. Afsar, M. Riaz, and M. Arif, "A comparison of baseline removal algorithms for electrocardiogram (ECG) based automated diagnosis of coronory heart disease," in *IEEE Proceedings of the Bioinformatics and Biomedical Engineering*. IEEE, 2009, pp. 1–4.

[161] E. Meijering, "A chronology of interpolation: From ancient astronomy to modern signal and image processing," *Proceedings of the IEEE*, vol. 90, no. 3, pp. 319–342, 2002.

[162] E. Waring, "Problems concerning interpolations." *Philosophical Transactions of the Royal Society London*, vol. 69, pp. 59–67, 1779.

[163] J. L. Lagrange, *Leçons élémentaires sur les Mathématiques, données à l'École normale, en 1795*, 1812.

[164] C. Runge, "Über empirische funktionen und die interpolation zwischen äquidistanten ordinaten," *Zeitschrift für Mathematik und Physik*, vol. 46, no. 224-243, p. 20, 1901.

[165] C. De Boor, *A practical guide to splines*. Springer-Verlag New York, 1978, vol. 27.

[166] G. E. Forsythe, M. A. Malcolm, and C. B. Moler, "Computer methods for mathematical computations," 1977.

[167] J. C. Beatty and B. A. Barsky, *An introduction to splines for use in computer graphics and geometric modeling*. Morgan Kaufmann, 1995.

[168] C. B. Moler, *Numerical Computing with MATLAB: Revised Reprint*. Siam, 2008.

[169] I. J. Schoenberg and I. J. Schoenberg, *Cardinal spline interpolation*. SIAM, 1973, vol. 12.

[170] T. I. Laakso, V. Valimaki, M. Karjalainen, and U. K. Laine, "Splitting the unit delay FIR/all pass filters design," *IEEE Signal Processing Magazine*, vol. 13, no. 1, pp. 30–60, 1996.

[171] Y. C. Eldar and A. V. Oppenheim, "Filterbank reconstruction of bandlimited signals from nonuniform and generalized samples," *IEEE Transactions on Signal Processing*, vol. 48, no. 10, pp. 2864–2875, 2000.

[172] R. S. Prendergast, B. C. Levy, and P. J. Hurst, "Reconstruction of band-limited periodic nonuniformly sampled signals through multirate filter banks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 51, no. 8, pp. 1612–1622, 2004.

[173] N. Chihab, A. Zergaïnoh, and J.-P. Astruc, "Generalized non-uniform B-spline functions for discrete signal interpolation," in *IEEE Proceedings of the Signal Processing and Its Applications*, vol. 2. IEEE, 2003, pp. 129–132.

[174] M. Obata, K. Mori, M. Ohira, K. Wada, and K. Toraichi, "Nonuniform sampling and interpolation based on partially dilated sampling function," in *IEEE Proceedings of the Communications, Computers and Signal Processing*, vol. 1. IEEE, 1997, pp. 453–456.

[175] C. Cenker, H. G. Feichtinger, and M. Herrmann, "Iterative algorithms in irregular sampling: A first comparison of methods," in *IEEE Proceedings of the Computers and Communications*. IEEE, 1991, pp. 483–489.

[176] F. Marvasti, M. Analoui, and M. Gamshadzahi, "Recovery of signals from nonuniform samples using iterative methods," *IEEE Transactions on Signal Processing*, vol. 39, no. 4, pp. 872–878, 1991.

[177] R. G. Wiley, "Recovery of bandlimited signals from unequally spaced samples," *IEEE Transactions on Communications*, vol. 26, no. 1, pp. 135–137, 1978.

[178] O. Guven, A. Eftekhar, W. Kindt, and T. Constandinou, "Computationally-efficient real-time interpolation algorithm for non-uniform sampled biosignals," *Healthcare Technology Letters*, 2016.

[179] J. C. Butcher, "A history of Runge-Kutta methods," *Applied Numerical Mathematics*, vol. 20, no. 3, pp. 247–260, 1996.

[180] P. K. Stein, M. S. Bosner, R. E. Kleiger, and B. M. Conger, "Heart rate variability: A measure of cardiac autonomic tone," *American Heart Journal*, vol. 127, no. 5, pp. 1376–1381, 1994.

[181] E. L. Melanson, "Resting heart rate variability in men varying in habitual physical activity." *Medicine and Science in Sports and Exercise*, vol. 32, no. 11, pp. 1894–1901, 2000.

[182] S. Seiler, O. Haugen, and E. Kuffel, "Autonomic recovery after exercise in trained athletes: Intensity and duration effects," *Medicine and Science in Sports and Exercise*, vol. 39, no. 8, p. 1366, 2007.

[183] W. F. Ganong and K. E. Barrett, *Review of medical physiology (24th ed.)*. McGraw-Hill Medical, 2012.

[184] C. P. Bonafide, P. W. Brady, R. Keren, P. H. Conway, K. Marsolo, and C. Daymont, "Development of heart and respiratory rate percentile curves for hospitalized children," *Pediatrics*, vol. 131, no. 4, pp. e1150–e1157, 2013.

[185] D. H. Barlow, P. M. Lehrer, R. L. Woolfolk, and W. E. Sime, *Principles and practice of stress management.* Guilford Press, 2007.

[186] V. Magagnin, M. Mauri, P. Cipresso, L. Mainardi, E. N. Brown, S. Cerutti, M. Villamira, and R. Barbieri, "Heart rate variability and respiratory sinus arrhythmia assessment of affective states by bivariate autoregressive spectral analysis," in *IEEE Proceedings of the Computing in Cardiology.* IEEE, 2010, pp. 145–148.

[187] D. Zuras, M. Cowlishaw, A. Aiken, M. Applegate, D. Bailey, S. Bass, D. Bhandarkar, M. Bhat, D. Bindel, S. Boldo *et al.*, "IEEE standard for floating-point arithmetic," *IEEE Std 754-2008*, pp. 1–70, 2008.

[188] American Heart Association, *Heart and stroke facts.* The Association, 1993.

[189] K. Fox, J. S. Borer, A. J. Camm, N. Danchin, R. Ferrari, J. L. L. Sendon, P. G. Steg, J.-C. Tardif, L. Tavazzi, and M. Tendera, "Resting heart rate in cardiovascular disease," *Journal of the American College of Cardiology*, vol. 50, no. 9, pp. 823–830, 2007.

[190] W. H. Press, *Numerical recipes 3rd edition: The art of scientific computing.* Cambridge university press, 2007.

[191] Texas Instruments. (2010, November) Keystone architecture universal asynchronous receiver/transmitter (UART). [Online]: http://www.ti.com.cn/cn/lit/ug/sprugp1/sprugp1.pdf

[192] Texas Instruments. (2016, May) MSP430FR58xx, MSP430FR59xx, MSP430FR68xx, and MSP430FR69xx family. [Online]: http://www.ti.com/lit/ug/slau367j/slau367j.pdf

[193] K. Venkat. (2006, September) Efficient multiplication and division using MSP430. [Online]: http://www.ti.com/lit/an/slaa329/slaa329.pdf

[194] C. Hamacher, Z. Vranesic, and S. Zaky, *Computer organization.* McGraw-Hill, 2002.

[195] Olimex. (2014, June) Shield-EKG-EMG bio-feedback shield user's manual. [Online]: https://www.olimex.com/Products/Duino/Shields/SHIELD-EKG-EMG/resources/SHIELD-EKG-EMG.pdf

[196] M. Mitchell. (2004, June) Choosing an ultra low-power MCU. [Online]: http://www.ti.com/lit/an/slaa207/slaa207.pdf

Appendix A

Publications

Published conference and letter materials constitute relevant sections of this thesis. The list of publications are listed as follows:

Chapter 3 contains content from BIOCAS 2014 conference publication:

- Guven, Onur, Amir Eftekhar, Reza Hoshyar, Giovanni Frattini, Wilko Kindt, and Timothy G. Constandinou. "Realtime ECG baseline removal: An isoelectric point estimation approach." In IEEE Proceedings of the Biomedical Circuits and Systems Conference (BioCAS), 2014 IEEE, pp. 29-32. IEEE, 2014.

Chapter 4 contains content from HTL 2016 letter publication:

- Guven, Onur, Amir Eftekhar, Wilko Kindt, and Timothy Constandinou. "Computationally-efficient realtime interpolation algorithm for non-uniform sampled biosignals" Healthcare Technology Letters (2016).

Appendix B

ECG Baseline Removal Algorithm

B.1. Differentiator Analysis

As discussed in Chapter 3, Eq. 3.10 covers N-point FIR approximated differentiator implementations. The response of each computationally efficient differentiator with integer coefficients is investigated and their parametric analysis plots are illustrated in Fig. B.1, B.2 and B.3.

Here, the sampling frequency is defined based on downsampling rate of 3 at 360 Hz and plots are generated in MATLAB. As can be seen, increasing the order N, also increases the total number of side lobes and their cut-off frequency location is defined based on the coefficient relationships. Increasing the differentiator order, N, however, limits its applications in ECG systems as the QRS complex is filtered partially with higher order differentiators. For a 7-point differentiator, the first sidelobe ranges approximately from 150 rad/s to 200 rad/s. This corresponds to 24 - 32 Hz and as the signal of interest that is required to be preserved extends to 30 Hz, a 7^th order increases the computational complexity and degrades system performance.

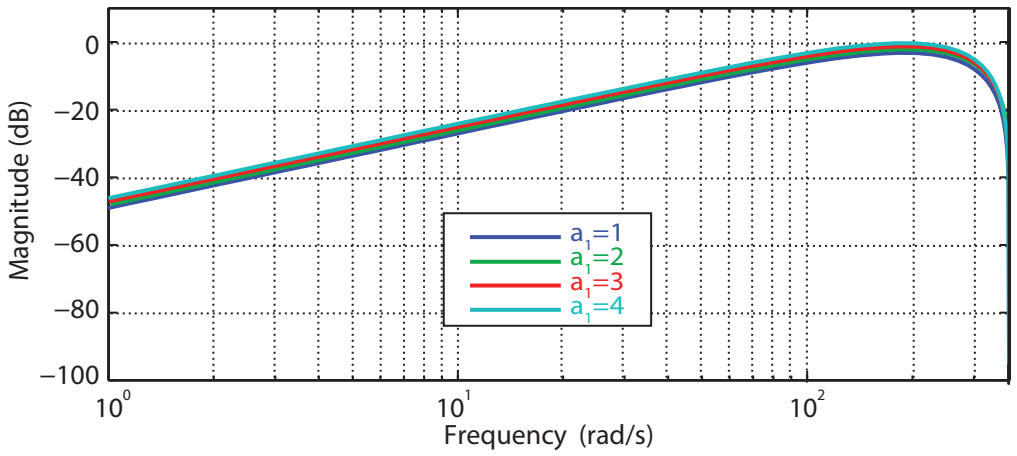


Figure B.1.: 3-point differentiator parametric analysis with varying a_1

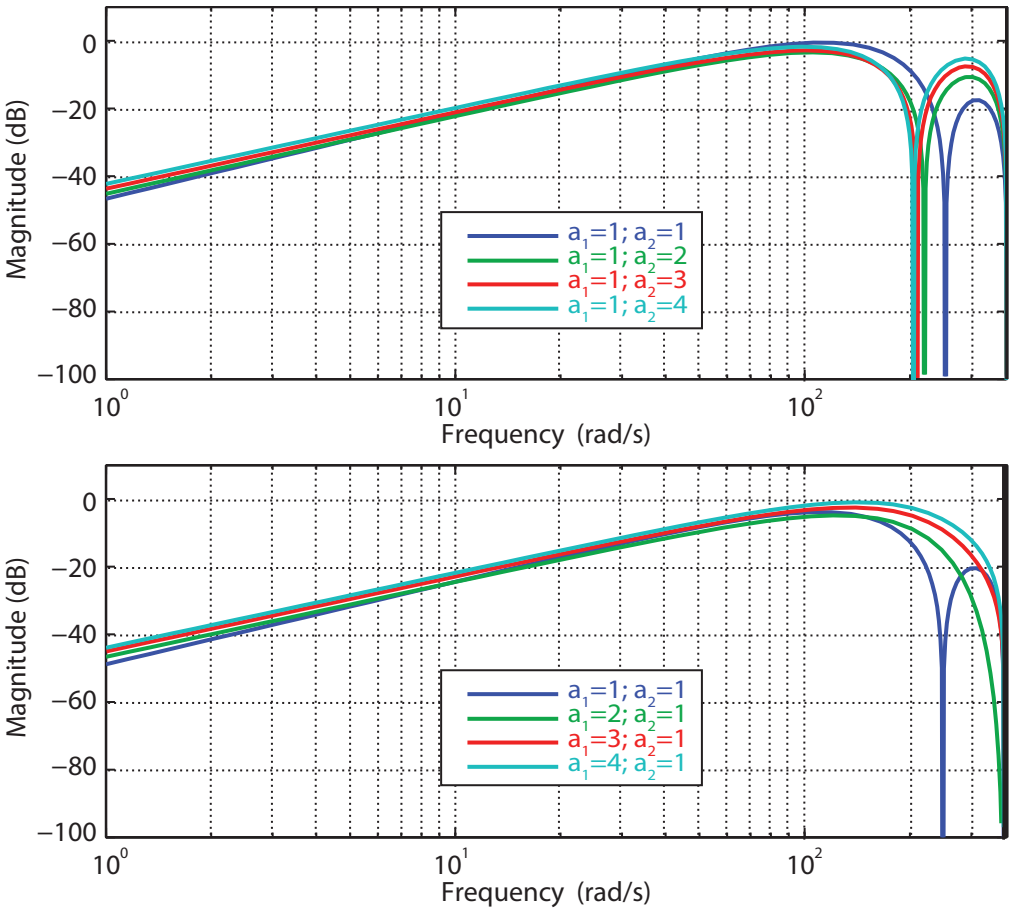


Figure B.2.: 5-point differentiator parametric analysis with varying a_1 and a_2

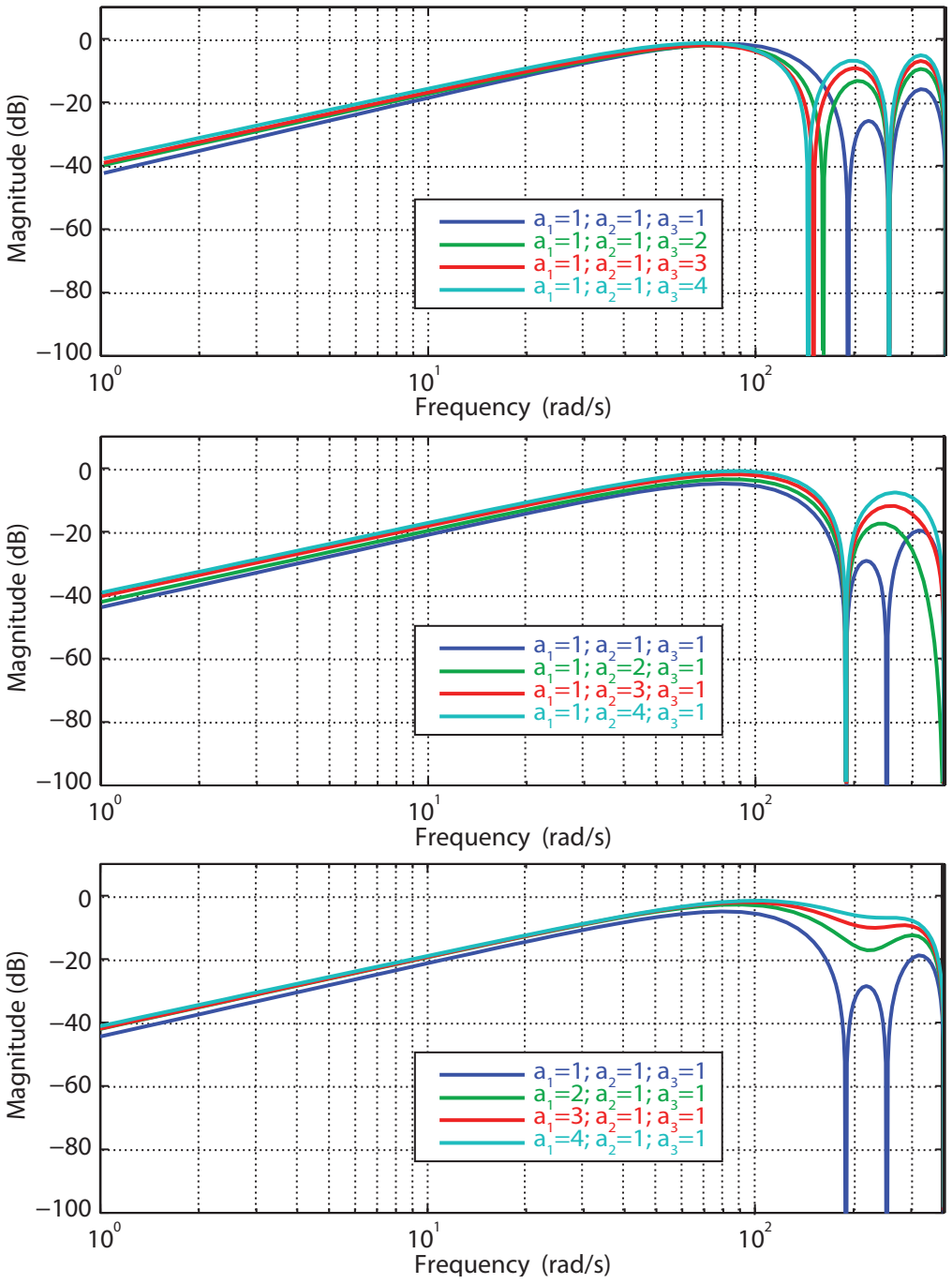


Figure B.3.: 7-point differentiator parametric analysis with varying a_1, a_2 and a_3

Appendix C

Interpolation Methods

C.1. WPL Interpolation Supplementary Equations & Figures

C.1.1. Equations

Other weighted mean equations that are tested apart from weighted arithmetic mean (WAM) approach are shown in Table C.1. Methods are denoted as: (1) Weighted geometric mean (WGM); (2) Weighted harmonic mean (WHM); (3) Weighted quadratic mean (WQM); (4) Weighted Heronian Mean (WHeM)

Table C.1.: Other WPL implementations

# WPL Imp.	Parameter a & b	Parameter c	H_{i_1}	H_{i_2}	H_{i_3}
WGM	$\frac{1}{2}, 1, \frac{3}{2}, 2$	1	$\sqrt[a+b]{M_{i-1}^{a} * M_{i}^{b}}$	$c * M_i$	$(3-c) * M_i - H_{i_1}$
WHM	$\frac{1}{2}, 1, \frac{3}{2}, 2$	1	$\frac{a+b}{\frac{a}{M_{i-1}} + \frac{b}{M_i}}$	$c * M_i$	$(3-c) * M_i - H_{i_1}$
WQM	$\frac{1}{2}, 1, \frac{3}{2}, 2$	1	$\sqrt{\frac{a * M_{i-1}^2 + b * M_i^2}{a+b}}$	$c * M_i$	$(3-c) * M_i - H_{i_1}$
WHeM	$\frac{1}{2}, 1, \frac{3}{2}, 2$	1	$\frac{2}{3} * WHM + \frac{1}{3} * WGM$	$c * M_i$	$(3-c) * M_i - H_{i_1}$

C.1.2. Figures

Similarly, these methods are tested on $1\,\mathrm{mV}_{p-p}$ synthetic sinusoidal signals and the figures shown here cover only $c = 1$ test results with the same turning point condition requirements. In these plots, same colour mapping has been utilised with the weighted mean average plots to identify accuracy results easily.

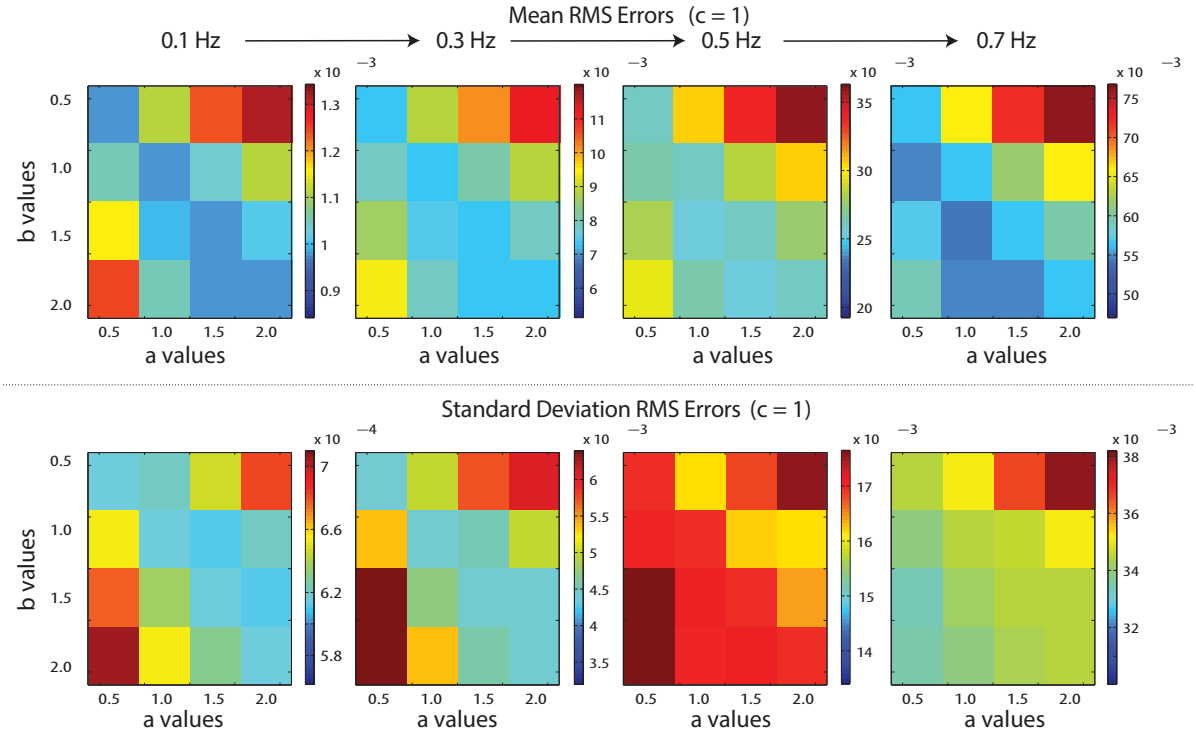


Figure C.1.: WGM based WPL interpolation equation, H_{i_1}, parametric analysis

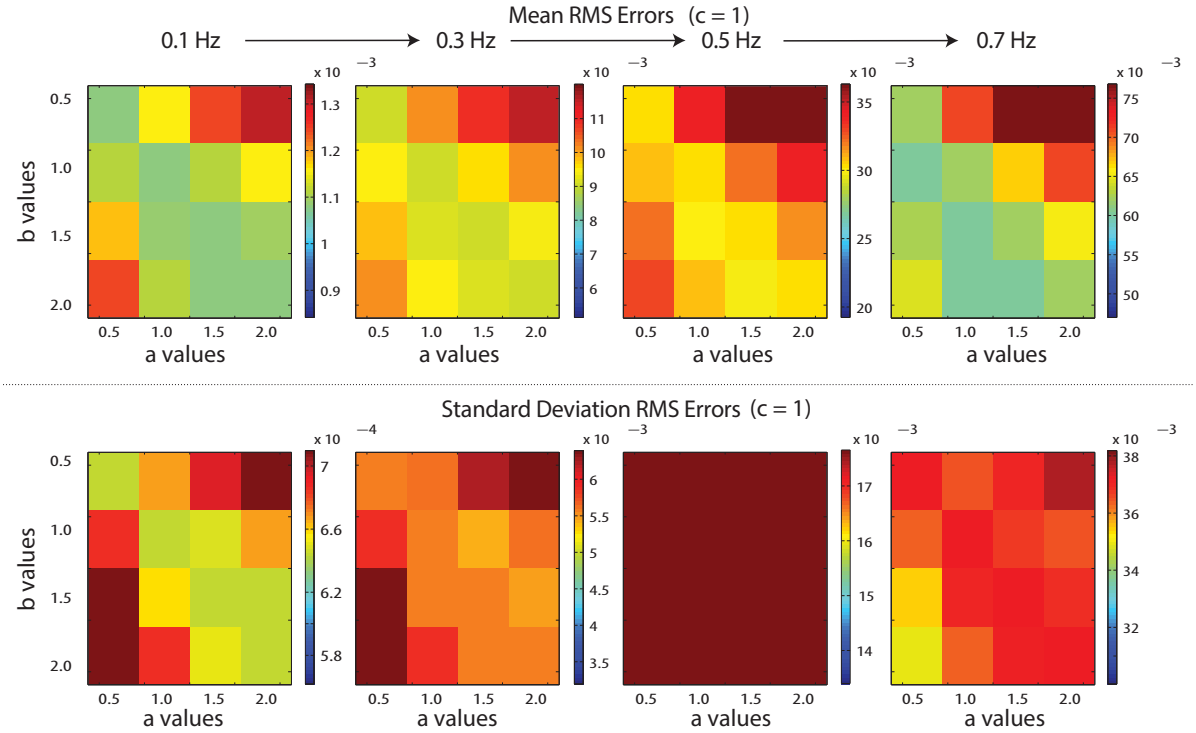


Figure C.2.: WHM based WPL interpolation equation, H_{i_1}, parametric analysis

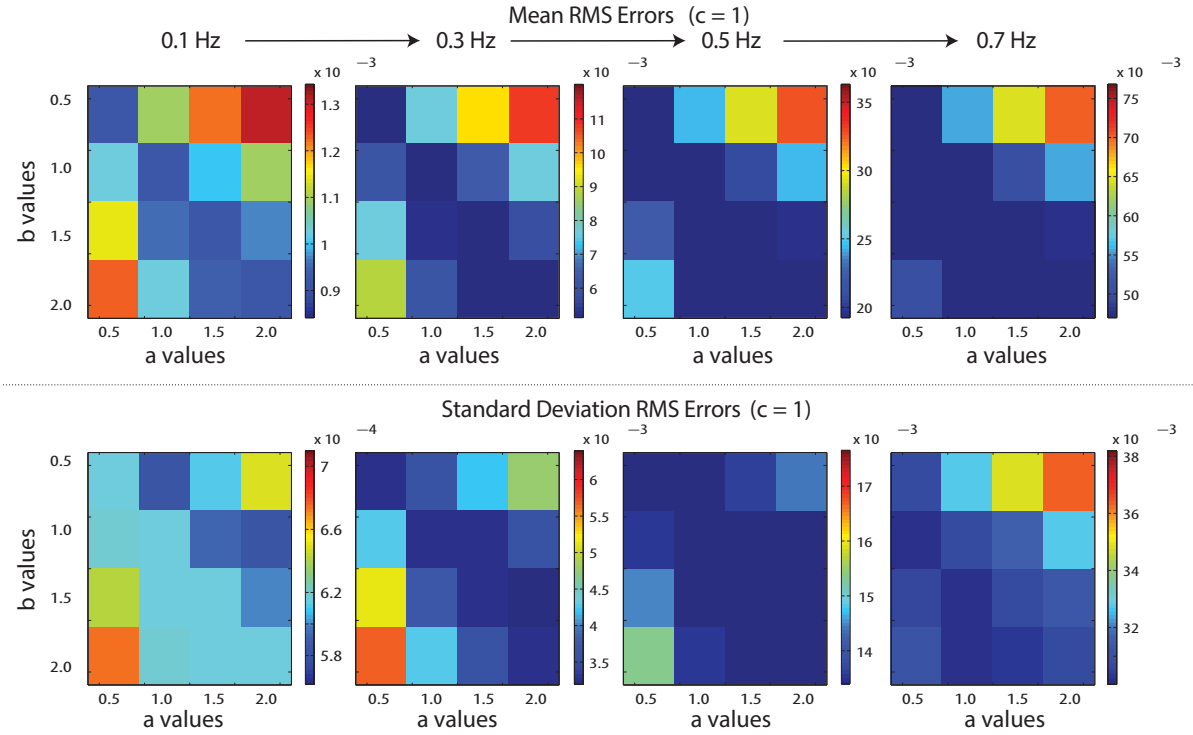


Figure C.3.: WQM based WPL interpolation equation, H_{i_1}, parametric analysis

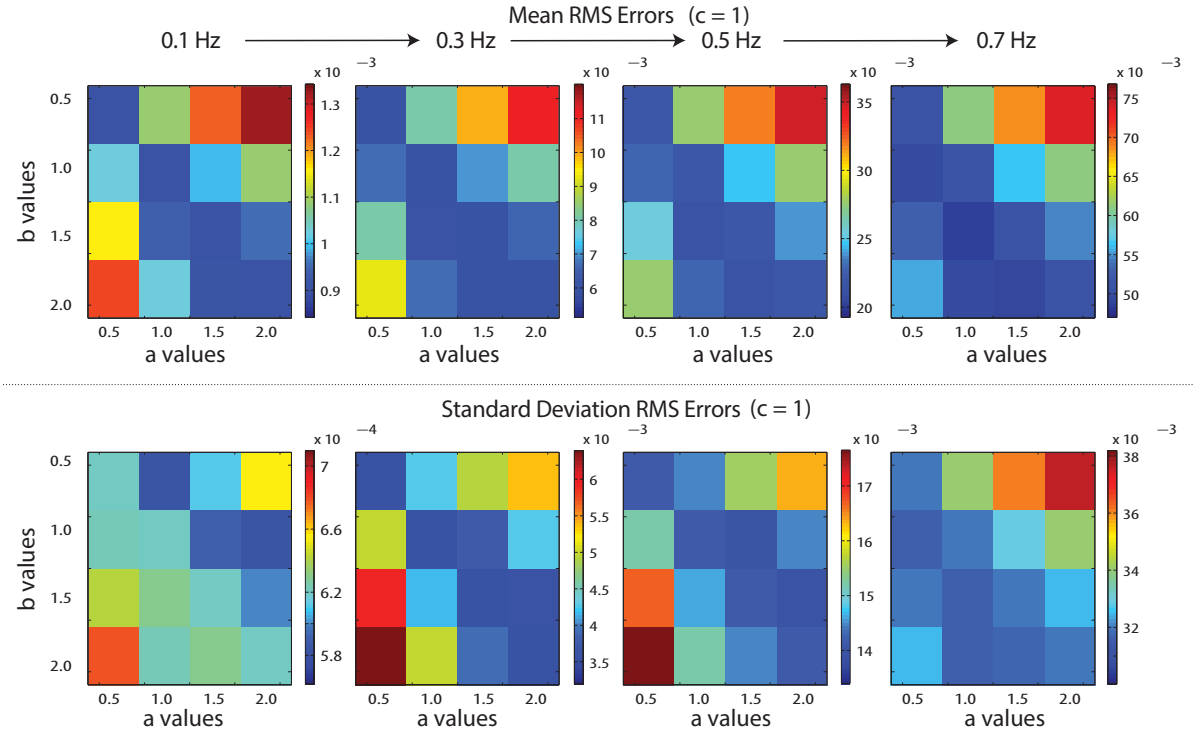


Figure C.4.: WHeM based WPL interpolation equation, H_{i_1}, parametric analysis

C.2. Maple Analytical Expressions

C.2.1. Linear Interpolation

Let, (x_1, y_1) and (x_2, y_2) be two interpolation points on a sinusoid function $f(x) = sin(x)$, linear equation of a line is written as follows:

$$f_{Linear}(x) = \left(\frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1}\right)(x - x_1) + \sin(x_1) \tag{C.1}$$

Integrating the difference of this sinusoid and the linear function yields an error function evaluated at x_2 and x_1 as in Eq.C.2:

$$Err_{Linear} = \int_{x_1}^{x_2} f(x) - f_{Linear}(x)\, dx$$

$$Err_{Linear} = -\cos(x_2) - \left(\frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1}\right)\left(1/2\, {x_2}^2 - x_1\, x_2\right) - \sin(x_1)\, x_2 + ... \tag{C.2}$$

$$\cos(x_1) - 1/2\left(\frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1}\right){x_1}^2 + \sin(x_1)\, x_1$$

C.2.2. Weighted Piecewise Linear Interpolation

Similarly, let, (x_0, y_0), (x_1, y_1) and (x_2, y_2) be three interpolation points on the same sinusoid function, $f(x) = sin(x)$, WPL interpolation equation of three equal segments are written as follows:

$$
\begin{aligned}
f_{WPL-S1}(x) = \quad & 1/2 \left(\frac{\sin(x_1) - \sin(x_0)}{x_1 - x_0} + \frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1} \right) (x - x_1) + \sin(x_1) \\
f_{WPL-S2}(x) = \quad & \left(\frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1} \right) (x - x_{1\_1}) + ... \\
& 1/2 \left(\frac{\sin(x_1) - \sin(x_0)}{x_1 - x_0} + \frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1} \right) (x_{1\_1} - x_1) + \sin(x_1) \\
f_{WPL-S3}(x) = & -1/2 \left(\frac{\sin(x_1) - \sin(x_0)}{x_1 - x_0} - 3 * \frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1} \right) (x - x_2) + \sin(x_2)
\end{aligned}
\tag{C.3}
$$

Integrating the difference of this sinusoid and the WPL functions yields an overall error function evaluated: (1) from x_1 to x_{1_1} for the first segment, f_{WPL-S1}, (2) x_{1_1} to x_{1_2} for the second segment, f_{WPL-S2} and (3) x_{1_2} to x_2 for the last segment, f_{WPL-S3} as shown below in Eq.C.5. As mentioned in Section 4.6.1, the interpolation points from x_0 to $x1$ is considered equal to the distance from x_1 to x_2 and this segment is partitioned into 3 equal smaller segments.

$$
\begin{aligned}
\sqrt{(x_1 - x_0)^2} &= \sqrt{(x_2 - x_1)^2} \\
x_{1_1} &= \frac{|x_2 - x_1|}{3} + x_1 \\
x_{1_2} &= \frac{2 * |x_2 - x_1|}{3} + x_1
\end{aligned}
\tag{C.4}
$$

$$Err_{WPL} = \int_{x_1}^{x_{1_1}} |f(x) - f_{S1}(x)|\,dx + \int_{x_{1_1}}^{x_{1_2}} |f(x) - f_{S2}(x)|\,dx + \int_{x_{1_2}}^{x_2} |f(x) - f_{S3}(x)|\,dx$$

$$| - \cos(x_2/3 + 2/3\,x_1) - 1/2\,(\frac{\sin(x_1) - \sin(2\,x_1 - x_2)}{x_2 - x_1} + \frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1}) * ...$$

$$(1/2\,(x_2/3 + 2/3\,x_1)^2 - x_1\,(x_2/3 + 2/3\,x_1)) - \sin(x_1)(x_2/3 + 2/3\,x_1) + \cos(x_1) - ...$$

$$1/4\,(\frac{\sin(x_1) - \sin(2\,x_1 - x_2)}{x_2 - x_1} + \frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1}){x_1}^2 + \sin(x_1)x_1| + ...$$

$$| - \cos(2/3\,x_2 + x_1/3) + \sin(x_1)(x_2/3 + 2/3\,x_1) - ...$$

$$\frac{(\sin(x_2) - \sin(x_1))(1/2\,(2/3\,x_2 + x_1/3)^2 - (x_2/3 + 2/3\,x_1)(2/3\,x_2 + x_1/3))}{x_2 - x_1} - ...$$

$$1/2\,(\frac{\sin(x_1) - \sin(2\,x_1 - x_2)}{x_2 - x_1} + \frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1})(x_2/3 - x_1/3)(2/3\,x_2 + x_1/3) - ...$$

$$\sin(x_1)(2/3\,x_2 + x_1/3) + \cos(x_2/3 + 2/3\,x_1) - 1/2\,\frac{(\sin(x_2) - \sin(x_1))(x_2/3 + 2/3\,x_1)^2}{x_2 - x_1}$$

$$+ 1/2\,(\frac{\sin(x_1) - \sin(2\,x_1 - x_2)}{x_2 - x_1} + \frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1})(x_2/3 - x_1/3)(x_2/3 + 2/3\,x_1)| + ...$$

$$| - \cos(x_2) + 1/4\,(3\,\frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1} - \frac{\sin(x_1) - \sin(2\,x_1 - x_2)}{x_2 - x_1}){x_2}^2 - \sin(x_2)x_2 + ...$$

$$\cos(2/3\,x_2 + x_1/3) + 1/2\,(3\,\frac{\sin(x_2) - \sin(x_1)}{x_2 - x_1} - \frac{\sin(x_1) - \sin(2\,x_1 - x_2)}{x_2 - x_1})$$

$$(1/2\,(2/3\,x_2 + x_1/3)^2 - x_2\,(2/3\,x_2 + x_1/3)) + \sin(x_2)(2/3\,x_2 + x_1/3)|$$

(C.5)

Appendix D

C Code: Real-Time ECG Baseline Wander Removal Algorithm

```
1  //  main.c
2  //  Created by OnurG on 12/08/2015.
3  //  Copyright (c) 2015 OnurG. All rights reserved.
4
5  #include <stdio.h>
6  #include "filterstage.h"
7  #include "Pan_Tompkins.h"
8  #include "Fiducial_Point_Detect.h"
9  #include "interpolation.h"
10
11 int main(int argc, const char * argv[]) {
12     char inputy_file[]="...", output_file[]="...", output_file2[]="...",
       output_file3[]="...";
13     int i=0, counter=0, counter2=0, counter3=0, same_distance_flag=0;
14     float original_inputy, filtered_inputy, moving_integrator_inputy,
       interpolator_inputx[3]={0}, interpolator_inputy[3]={0}, output, output2,
       output3 fiducial_point_inputx=0, fiducial_point_inputy=0,processed_signal=0;
15     float filteredinput[12]={0}, unfilteredinput[8]={0};
16
17     FILE *open_inputy_file= fopen(inputy_file,"r");        //input files to read
18     FILE *open_output_file= fopen(output_file,"wb");       //output files to write
19     FILE *open_output_file2= fopen(output_file2,"wb");     //output files to write
20     FILE *open_output_file3= fopen(output_file3,"wb");     //output files to write
21
22     while(1){
23
24         fscanf(open_inputy_file, "%f", &original_inputy);
25         unfilteredinput[counter2]=original_inputy;
26         if (counter==3){
27             filter_stage_level(&original_inputy,&output);
28             filtered_inputy=output;
29             filteredinput[counter3]=filtered_inputy;
30             Pan_Tompkins(&filtered_inputy,&output);
```

```
31              moving_integrator_inputy=output;
32              fiducial_point_detect(&unfilteredinput[counter2],&filteredinput[
    counter3],&moving_integrator_inputy,&fiducial_point_inputx,&
    fiducial_point_inputy,&output2,&output3,counter2,counter3);
33              fprintf(open_output_file2,"%f\n",output2);
34              fprintf(open_output_file3,"%f\n",output3);
35              if (interpolator_inputx[2]!=fiducial_point_inputx){
36                  for (i=0;i<2;i++){
37                      interpolator_inputx[i]=interpolator_inputx[i+1];
38                      interpolator_inputy[i]=interpolator_inputy[i+1];
39                      }
40                  interpolator_inputx[2]=fiducial_point_inputx;
41                  interpolator_inputy[2]=fiducial_point_inputy;
42                  if (interpolator_inputx[2]-interpolator_inputx[1]==
    interpolator_inputx[1]-interpolator_inputx[0]){
43                      same_distance_flag=1;
44                  }
45              }
46              else {
47              interpolator_inputx[2]=fiducial_point_inputx;
48              interpolator_inputy[2]=fiducial_point_inputy;
49              }
50              counter=0;
51              counter3++;
52          }
53          counter++;
54          counter2++;
55          if (counter2==8)
56              counter2=0;
57          if (counter3==12)
58              counter3=0;
59          if (interpolator_inputx[0]>0){
60              hybrid_interpolation(&interpolator_inputx[2], &interpolator_inputy
    [2],&processed_signal,&same_distance_flag);
61          }
62          fprintf(open_output_file,"%f\n",processed_signal);
63      }
64      fclose(open_inputy_file);                          //input file to close
65      fclose(open_output_file);                          //output file to close
66      fclose(open_output_file2);                         //output file to close
67      fclose(open_output_file3);                         //output file to close
68      printf("Completed\n");
69      return 0;
70  }
```

```c
//  filterstage.c
//  Created by OnurG on 12/08/2015.
//  Copyright (c) 2015 OnurG. All rights reserved.

#include "filterstage.h"


void filter_stage_level(float *pInput, float *pOutput){

    static float delay_cells_high_pass[4];
    static float delay_cells_low_pass[4];

    float filter_coefficients[24] =                 //Scaled for floating point
    {
        b0, b1, a1, b2, a2
        1,-2,-1.9512939453125, 1, 0.954833984375,   //highpass_first_biquad
        1,-2,-1.89093017578125,1,0.89434814453125,  //highpass_second_biquad
        1, 2, 0.9725341796875, 1,0.54205322265625,  //lowpass_first_biquad
        1, 2, 0.73468017578125,1,0.16485595703125,  //lowpass_second_biquad
        0.25, 0.25, 0.25, 0.25                      // moving_average filter
    };

    filter_pointers Progress;
    Progress.pInput = pInput;
    Progress.pOutput = pOutput;
    Progress.pCoefficients=filter_coefficients;
    Progress.pDelays_H=delay_cells_high_pass;
    Progress.pDelays_L=delay_cells_low_pass;

    Progress.stage=0;
    filter_biquad_highpass (&Progress);      // Running HIGH-PASS first biquad
    Progress.pInput=Progress.pOutput;        // Assigning the output to input

    Progress.stage=1;
    filter_biquad_highpass (&Progress);      // Running HIGH-PASS second biquad
    Progress.pInput=Progress.pOutput;        // Assigning the output to input

    Progress.stage=0;
    filter_biquad_lowpass (&Progress);       // Running LOW-PASS first biquad
    Progress.pInput=Progress.pOutput;        // Assigning the output to input

    Progress.stage=1;
    filter_biquad_lowpass (&Progress);      // Running LOW-PASS second biquad
    Progress.pInput=Progress.pOutput;       // Assigning the output to input

    filter_moving_average(&Progress);       // Running MOVING AVERAGE

    Progress.pCoefficients=filter_coefficients; // Restarting Coefficients

}
```

```
51
52 void filter_biquad_highpass(filter_pointers *biquad){  // HIGH PASS FILTER
53
54     float *pInput= biquad->pInput;
55     float inputtemp;
56     float *pOutput= biquad->pOutput;
57
58     float *temp1=(biquad->pDelays_H);          // Reading Delay Cell registers
59     float d1_stage1=*(biquad->pDelays_H++);
60     float d2_stage1=*(biquad->pDelays_H++);
61     float *temp2=(biquad->pDelays_H);
62     float d1_stage2=*(biquad->pDelays_H++);
63     float d2_stage2=*(biquad->pDelays_H);
64
65     float b0=*(biquad->pCoefficients++);     // Reading Coefficient registers
66     float b1=*(biquad->pCoefficients++);
67     float a1=*(biquad->pCoefficients++);
68     float b2=*(biquad->pCoefficients++);
69     float a2=*(biquad->pCoefficients++);
70
71     if (biquad->stage==0)                    // biquad implementation -> STAGE 1
72     { // b0 b1 b2 a1 a2 are filter coefficients - d1,d2 are delay cells
73         biquad->pDelays_H=temp1;
74         *pOutput = b0 * *pInput + d1_stage1;
75         *biquad->pDelays_H++ = b1 * *pInput - a1 * *pOutput + d2_stage1;
76         *biquad->pDelays_H = b2 * *pInput - a2 * *pOutput;
77         biquad->pDelays_H=temp1;
78     }
79     else if (biquad->stage==1)               // biquad implementation -> STAGE 2
80     { // b0 b1 b2 a1 a2 are filter coefficients - d1,d2 are delay cells
81         inputtemp=*pInput;
82         biquad->pDelays_H=temp2;
83         *pOutput = b0 * inputtemp + d1_stage2;
84         *biquad->pDelays_H++= b1 * inputtemp - a1 * *pOutput + d2_stage2;
85         *biquad->pDelays_H = b2 * inputtemp - a2 * *pOutput;
86         biquad->pDelays_H=temp1;
87     }
88 }
89
90 void filter_biquad_lowpass(filter_pointers *biquad){     //LOW PASS FILTER
91
92     float *pInput= biquad->pInput;
93     float inputtemp;
94     float *pOutput= biquad->pOutput;
95
96     float *temp1=(biquad->pDelays_L);          // Reading Delay Cell registers
97     float d1_stage1=*(biquad->pDelays_L++);
98     float d2_stage1=*(biquad->pDelays_L++);
99     float *temp2=(biquad->pDelays_L);
100    float d1_stage2=*(biquad->pDelays_L++);
```

```
101     float d2_stage2=*(biquad->pDelays_L);
102
103     float b0=*(biquad->pCoefficients++);       // Reading Coefficient registers
104     float b1=*(biquad->pCoefficients++);
105     float a1=*(biquad->pCoefficients++);
106     float b2=*(biquad->pCoefficients++);
107     float a2=*(biquad->pCoefficients++);
108
109
110     if (biquad->stage==0)          // biquad implementation -> STAGE 1
111     { // b0 b1 b2 a1 a2 are filter coefficients - d1,d2 are delay cells
112         inputtemp=*pInput;
113         biquad->pDelays_L=temp1;
114         *pOutput = b0 * inputtemp + d1_stage1;
115         *biquad->pDelays_L++ = b1 * inputtemp - a1 * *pOutput + d2_stage1;
116         *biquad->pDelays_L = b2 * inputtemp - a2 * *pOutput;
117         biquad->pDelays_L=temp1;
118     }
119     else if (biquad->stage==1)         // biquad implementation -> STAGE 2
120     { // b0 b1 b2 a1 a2 are filter coefficients - d1,d2 are delay cells
121         inputtemp=*pInput;
122         biquad->pDelays_L=temp2;
123         *pOutput = b0 * inputtemp + d1_stage2;
124         *biquad->pDelays_L++= b1 * inputtemp - a1 * *pOutput + d2_stage2;
125         *biquad->pDelays_L = b2 * inputtemp - a2 * *pOutput;
126         biquad->pDelays_L=temp1;
127     }
128 }
129
130 void filter_moving_average(filter_pointers *moving){  //MOVING AVERAGE FILTER
131     int i;
132     static float inputtemp[4];
133     float *pInput= moving->pInput;
134     float *pOutput= moving->pOutput;
135
136     float b0=*(moving->pCoefficients++);      // Reading Coefficient registers
137     float b1=*(moving->pCoefficients++);
138     float b2=*(moving->pCoefficients++);
139     float b3=*(moving->pCoefficients++);
140
141     inputtemp[3]=*pInput;
142     *pOutput = b0 * inputtemp[3] + b1*  inputtemp[2] + b2 * inputtemp[1]  + b3 *
        inputtemp[0];                  // Moving Average Filter Implementation
143
144     for (i=0; i<3; i++)                        // Updating Delay Registers
145         inputtemp[i] = inputtemp[i+1];
146
147 }
```

```c
//  Pan_Tompkins.c
//  Created by OnurG on 16/10/2015.
//  Copyright (c) 2015 OnurG. All rights reserved.

#include "Pan_Tompkins.h"

void Pan_Tompkins(float *pInput, float *pOutput){

    int i;
    float y;
    static float x_derv[4]={0}, x_integral[8]={0};
    static float sum=0;
    static int ptr=0;

    Pan_Tompkins_pointers Pan;
    Pan.pInput= pInput;
    Pan.pOutput= pOutput;

    //--------------------Differentiator Section----------------------//

    // 5 - point differentiator - 1/8 (2x(nT) + x(nT-T) - x(nT-3T) - 2x(nT-4T))
    y= (*Pan.pInput * 2 ) + x_derv[3] -x_derv[1] - (x_derv[0] * 2);
    y/=8;

    for (i=0; i<3; i++)                         // New sample shift
        x_derv[i] = x_derv [i+1];

    x_derv[3] = *Pan.pInput;                    // Assigning new input

    //--------------------Squaring Section----------------------------//

    y*=y;                                       // Squaring Operation

    //-----------------Moving Integrator Section--------------------//

    if (++ptr==8)                               // Loading the new sample space
        ptr=0;

    sum-=x_integral[ptr]/8;                     // Removing old sample

    x_integral[ptr]=y;                          // Updating Integrator

    sum+=y/8;                                   // Adding averaged new sample

    //----------------------Output Section----------------------------//
    *Pan.pOutput=sum;                           // Updating Output

}
```

```
1  //  Fiducial_Point_Detect.c
2  //  Created by OnurG on 16/10/2015.
3  //  Copyright (c) 2016 OnurG. All rights reserved.
4
5  #include <math.h>
6  #include "Fiducial_Point_Detect.h"
7
8  void fiducial_point_detect(float *pUnfilteredinput,float *pFilteredinput, float *
       pInput, float *pFiducial_point_inputx,float *pFiducial_point_inputy,float *
       pOutput2,float *pOutput3,int counter2,int counter3){
9
10 //--------------- Threshold Variables ------------------//
11     int i, QRS_flag=0;
12     static int count,count2,watchdog=0;
13     static float threshold,moving_integrator[3],previous_threshold,new_threshold;
14 //--------------- Fiducial Point Variables -------------//
15     static int Tsearch=0,Psearch=0,QRS_Detected=0, T_Detected=0, P_Detected=0,
       wait_window=0, search_window=0,timestamp=0,MTC_flag=0, T_threshold_counter=0,
       P_threshold_counter=0, discrepancy_counter1=0, discrepancy_counter2=0,
       discrepancy_flag=0, detected_beats1=0, detected_beats2=0;
16     static float Filteredinput[2], MTC_tempx, MTC_tempy, T_multiplier=1.0,
       P_multiplier=1.0, J2J1diff=0, J2J3diff=0, J2J1_locked=0, J2J3_locked=0;
17     float Unfilteredinput[5]={0}, j1=0, j2=0, j3=0;
18 //-------------------------------Updating Arrays-------------------------------//
19     timestamp++;        // Fiducial point in terms of sampling time - timestamp
20     for (i=4;i>=0;i--){                          // Updating Unfiltered input
21         Unfilteredinput[i]=*pUnfilteredinput;
22         if (counter2==0)
23             pUnfilteredinput=pUnfilteredinput+7; // For Correct Dereferencing
24         else
25             pUnfilteredinput--;
26         counter2--;
27     }
28     if (counter3<5)
29         Filteredinput[1]=*(pFilteredinput+7); // Updating Filteredinput array
30     else
31         Filteredinput[1]=*(pFilteredinput-5); // For Correct Dereferencing
32     moving_integrator[2]=*pInput;              // Updating moving integrator array
33 //--------------- Threshold Generation of QRS, P and T waves-------------------//
34     if (moving_integrator[0] < moving_integrator[1] && moving_integrator[1] >
       moving_integrator[2] && moving_integrator[1] > previous_threshold) {
35         new_threshold = 0.5 * previous_threshold + 0.125 * moving_integrator[1];
36         watchdog=0;        // Setting up the new threshold and resetting watchdog
37     }
38     watchdog++;              // Watchdog for system reset
39     if (watchdog == 180)
40         new_threshold=0;
41     if (moving_integrator[2]>new_threshold && moving_integrator [1]<new_threshold
       ){                       // QRS detection
42         QRS_flag=1;
```

```
43          count=1;
44      }
45      if (count > 0){          // QRS flag generation
46          QRS_flag=1;
47          count++;
48          if (count>15){
49              count=0;
50              count2=1;
51          }
52      }
53      if (QRS_flag==1){         // Adjusting Thresholds
54          threshold=new_threshold/32.0;
55      }
56      else if (count2<35){
57          threshold=new_threshold/128.0;
58          count2++;
59      }
60      else
61          threshold=new_threshold/312.0;
62  //------------------ Detection of QRS, P and T waves-------------------------//
63      // Checking overall signal for threshold crossings
64      if (Filteredinput[0]>threshold && Filteredinput[1]<threshold && QRS_flag==1){
65          QRS_Detected=1;
66          P_Detected=0;         // Forfeit any MTC related search
67          T_Detected=0;
68          wait_window=0;
69          search_window=0;
70          Psearch=0;
71          if (MTC_flag>0){
72              *pFiducial_point_inputx=MTC_tempx;  // Multiple P wave detection
73              j1=MTC_tempy;                  // Used for discrepancy calculations
74              *pFiducial_point_inputy=MTC_tempy+J2J1_locked;      // Discrepancy
75          }
76          MTC_flag=0;
77          if (T_threshold_counter==8)
              // If no T wave threshold crossing is detected, adjust threshold
78              T_multiplier=0.5;
79          T_threshold_counter++;
80      }
81      else if (Filteredinput[0]>(T_multiplier*threshold) && Filteredinput[1]<(
        T_multiplier*threshold) && Tsearch==1){
82          T_Detected=1;
83          T_threshold_counter=0;
84          wait_window=0;
85          search_window=0;
86          if (P_threshold_counter==8)
              // If no P wave threshold crossing is detected, adjust threshold
87              P_multiplier=0.5;
88          P_threshold_counter++;
89      }
```

```
90      else if (Filteredinput[0]>(P_multiplier*threshold) && Filteredinput[1]<(
        P_multiplier*threshold) && Psearch==1){
91          P_Detected=1;
92          P_threshold_counter=0;
93          wait_window=0;
94          search_window=0;
95      }
96  //------------------Detection of Fiducial Point Locations---------------------//
97      if (QRS_Detected==1){
98          if (wait_window > 6){
99              if ((Unfilteredinput[0] > Unfilteredinput[1] && Unfilteredinput[1] >
        Unfilteredinput[2] && search_window < 10) || (fabs(Unfilteredinput[0]-
        Unfilteredinput[1])<3 && search_window < 10)){
100                 *pFiducial_point_inputx= 3*timestamp;
101                 j2=Unfilteredinput[1];      // Used for discrepancy calculations
102                 *pFiducial_point_inputy=Unfilteredinput[1];
103                 QRS_Detected=0;
104                 Tsearch=1;
105             }
106             else if ((Unfilteredinput[1] > Unfilteredinput[2] && Unfilteredinput
        [2] > Unfilteredinput[3] && search_window < 10)|| (fabs(Unfilteredinput[1]-
        Unfilteredinput[2])<3 && search_window < 10)){
107                 *pFiducial_point_inputx= 3*timestamp;
108                 j2=Unfilteredinput[1];      // Used for discrepancy calculations
109                 *pFiducial_point_inputy=Unfilteredinput[1];
110                 QRS_Detected=0;
111                 Tsearch=1;
112             }
113             else if ((Unfilteredinput[2]>Unfilteredinput[3] && Unfilteredinput[3]
        > Unfilteredinput[4] && search_window < 10) || (fabs(Unfilteredinput[2]-
        Unfilteredinput[3])<3 && search_window < 10)){
114                 *pFiducial_point_inputx= 3*timestamp;
115                 j2=Unfilteredinput[1];    // Used for discrepancy calculations
116                 *pFiducial_point_inputy=Unfilteredinput[1];
117                 QRS_Detected=0;
118                 Tsearch=1;
119             }
120             search_window++;
121         }
122     wait_window++;
123     }
124     if (T_Detected==1){
125         if (wait_window > 8){
126             if ((Unfilteredinput[0] > Unfilteredinput[1] && Unfilteredinput[1] >
        Unfilteredinput[2] && search_window < 6) || (fabs(Unfilteredinput[0]-
        Unfilteredinput[1])<3 && search_window < 6)){
127                 *pFiducial_point_inputx= 3*timestamp;
128                 j3=Unfilteredinput[1];    // Used for discrepancy calculations
129                 *pFiducial_point_inputy=Unfilteredinput[1]+J2J3_locked;// Discr.
130                 T_Detected=0;
```

```
131                 Tsearch=0;
132                 Psearch=1;
133             }
134             else if ((Unfilteredinput[1] > Unfilteredinput[2] && Unfilteredinput
    [2] > Unfilteredinput[3] && search_window < 6)|| (fabs(Unfilteredinput[1]-
    Unfilteredinput[2])<3 && search_window < 6)){
135                 *pFiducial_point_inputx= 3*timestamp;
136                 j3=Unfilteredinput[1];    // Used for discrepancy calculations
137                 *pFiducial_point_inputy=Unfilteredinput[1]+J2J3_locked;// Discr.
138                 T_Detected=0;
139                 Tsearch=0;
140                 Psearch=1;
141             }
142             else if ((Unfilteredinput[2]>Unfilteredinput[3] && Unfilteredinput[3]
    > Unfilteredinput[4] && search_window < 6) || (fabs(Unfilteredinput[2]-
    Unfilteredinput[3])<3 && search_window < 6)){
143                 *pFiducial_point_inputx= 3*timestamp;
144                 j3=Unfilteredinput[1];   // Used for discrepancy calculations
145                 *pFiducial_point_inputy=Unfilteredinput[1]+J2J3_locked;// Discr.
146                 T_Detected=0;
147                 Tsearch=0;
148                 Psearch=1;
149             }
150             search_window++;
151         }
152         wait_window++;
153     }
154     if (P_Detected==1){
155         if (wait_window > 0){
156             if ((Unfilteredinput[0] > Unfilteredinput[1] && Unfilteredinput[1] >
    Unfilteredinput[2] && search_window < 6) || (fabs(Unfilteredinput[0]-
    Unfilteredinput[1])<3 && search_window < 6)){
157                 MTC_tempx= 3*timestamp;  //Temporarily storing multiple P waves
158                 MTC_tempy=Unfilteredinput[1];
159                 P_Detected=0;
160                 MTC_flag++;
161             }
162             else if ((Unfilteredinput[1] > Unfilteredinput[2] && Unfilteredinput
    [2] > Unfilteredinput[3] && search_window < 6)|| (fabs(Unfilteredinput[1]-
    Unfilteredinput[2])<3 && search_window < 6)){
163                 MTC_tempx= 3*timestamp  //Temporarily storing multiple P waves
164                 MTC_tempy=Unfilteredinput[1];
165                 P_Detected=0;
166                 MTC_flag++;
167             }
168             else if ((Unfilteredinput[2]>Unfilteredinput[3] && Unfilteredinput[3]
    > Unfilteredinput[4] && search_window < 6) || (fabs(Unfilteredinput[2]-
    Unfilteredinput[3])<3 && search_window < 6)){
169                 MTC_tempx= 3*timestamp; //Temporarily storing multiple P waves
170                 MTC_tempy=Unfilteredinput[1];
```

```
171                     P_Detected=0;
172                     MTC_flag++;
173                 }
174                 search_window++;
175             }
176             wait_window++;
177         }
178 //------------------------Discrepancy Calculations-----------------------------//
179         if (timestamp>350 && timestamp<1350 && discrepancy_flag==0){
180         // 8 consecutive heart cycles at 72bpm require 800 samples
181             if (j1>0)
182                 discrepancy_counter1++;
183             if (discrepancy_counter1 >0){
184                 J2J1diff+=j2-j1;            // j1,j2,j3 are float=0 when no detection
185                 if (discrepancy_counter1 >0 && j2>0) // Counts total detected beats
186                     detected_beats1++;
187             }
188             if (j2>0){
189                 discrepancy_counter1--;
190                     if (discrepancy_counter1<0)
191                         discrepancy_counter1=0;
192                 discrepancy_counter2++;
193             }
194             if (discrepancy_counter2 >0){
195                 J2J3diff+=j2-j3;                        // J2J1diff-J2J3diff are static
196                 if (discrepancy_counter2 >0 && j2>0) // Counts total detected beats
197                     detected_beats2++;
198             }
199             if (j3>0)
200                 discrepancy_counter2--;
201                     if (discrepancy_counter2<0)
202                         discrepancy_counter2=0;
203             if (timestamp>1150 && discrepancy_counter1==0 && discrepancy_counter2==0)
                {       // Averaging discrepancy calculations
204                 J2J1_locked=J2J1diff/detected_beats1;
205                 J2J3_locked=J2J3diff/detected_beats2;
206                 discrepancy_flag=1;
207             }
208         }
209 //-----------------Update Registries Prepare for next cycle--------------------//
210         for (i=0; i<2; i++){  // Moving integrator shift in the memory
211             moving_integrator[i] = moving_integrator[i+1];
212         }
213         Filteredinput[0]=Filteredinput[1];    // Filtered input in the memory
214         previous_threshold=new_threshold;    // Updating Threshold Registers
215         *pOutput2=threshold;
216         *pOutput3=*pFiducial_point_inputx;
217 }
```

```c
//  interpolation.c
//  Created by OnurG on 18/08/2015.
//  Copyright (c) 2015 OnurG. All rights reserved.

#include <math.h>
#include "interpolation.h"

void hybrid_interpolation(float *pInput_x, float *pInput_y, float *pOutput, int *pFlag){

    static float slope1[12],slope2[12],location_y[12],output;
    //slope1 & slope2 for WPL interpolation
    float tempy,temp_plus,temp_minus;;
    static int tempx,duration_temp={0},timestamp=0,delay=0;;
    static int duration[12]={0},duration_counter=0,counter=0,i=11;

    timestamp++;
    tempy=*pInput_y;
    //store missed points in the memory until interpolation is complete
    tempx=*pInput_x;
    location_y[i]=tempy;
    duration[i]=tempx-*(--pInput_x);// time frame for interpolation
    // second slope generation
    slope2[i]= (tempy-*(--pInput_y))/(tempx-*(pInput_x));
    tempy=*pInput_y;                // to avoid unsequenced operation by pointers
    tempx=*pInput_x;
    // first slope generation
    slope1[i]= (tempy-*(--pInput_y))/(tempx-*(--pInput_x));
    if ((duration[i]!=0 && duration_temp!=duration[i]) || *pFlag==1){
        duration_temp=duration[i];
        i--;
        *pFlag=0;
    }
    if (i<1)
        i=1;
//---------------------- Empty processed buffers ----------------------------//
    if (duration_counter==duration[11]){
        i++;
        duration_counter=0;
        counter=0;
        for (int j=11; j>1; j--) {
            duration[j]=duration[j-1];
            slope1[j]=slope1[j-1];
            slope2[j]=slope2[j-1];
            location_y[j]=location_y[j-1];
            if (duration[j]==0) {
                duration[j+1]=0;
                slope1[j+1]=0;
                slope2[j+1]=0;
                location_y[j+1]=0;
```

```
            }
        }
    }
//-------------------- Fill empty buffers and check----------------------//
    if (duration[11]==0){
        delay=0;
        i=11;
    }
    if (delay<300){
        output+=0;
        *pOutput=output;
    }
    if (delay>300){
        delay=300;
//-----------------------------WPL Interpolation-------------------------------//
        duration_counter++;
        if ((slope1[11]>0 && slope2[11]<0) || (slope1[11]<0 && slope2[11]>0) ||
    0.75*fabs(slope1[11])>fabs(slope2[11]) || fabs(slope1[11])<0.75*fabs(slope2
    [11])) {
            // slope that will be generated for segment 1
            temp_plus= (slope1[11]+slope2[11])/2;
            // slope that will be generated for segment 3
            temp_minus= (2*slope2[11]-temp_plus);
            counter++;
            if (counter<duration[11]/3){
                output+=temp_plus;
                *pOutput=output;
            }
            else if (counter >= duration[11]/3 && counter < duration[11]/3*2){
                output+=slope2[11];
                *pOutput=output;
            }
            else {
                output+=temp_minus;
                *pOutput=output;
            }
            if (duration_counter==duration[11]){
                output=location_y[11];
                *pOutput=output;
            }
        }
//-----------------------Linear Interpolation-------------------------------//
        else {
            output+=slope2[11];
            *pOutput=output;
        }
    }
    delay++;
}
```

Appendix E

CCS Code: Real-Time ECG Baseline Wander Removal Algorithm

```
//  main.c
//  Created by OnurG on 12/05/2016.
//  Copyright (c) 2015 OnurG. All rights reserved.

#include <msp430FR6989.h>
#include <stdio.h>
#include "filterstage.h"
#include "Pan_Tompkins.h"
#include "Fiducial_Point_Detect.h"
#include "interpolation.h"
//------------------------------------IO data-------------------------------------//
int inputy_MSB=0;
float inputy=0;
float output=0;
unsigned char *ptr;
//---------------------------- Function Definitions-----------------------------//
//Processing
void processing (float inputy, float *pInterpolation_Output);
//Microcontroller
void setClock(void);
void setUART(void);
void setMultiplier(void);
//---------------------------- Register Functions-----------------------------//
// Startup clock system with max DCO~16MHz
void setClock(void){
    CSCTL0_H = CSKEY >> 8;                          // Unlock clock registers
    CSCTL1 = DCOFSEL_4 | DCORSEL;                   // Set DCO to 16MHz
    CSCTL2 = SELA__VLOCLK | SELS__DCOCLK | SELM__DCOCLK; // Set Clocks
    CSCTL3 = DIVA__1 | DIVS__1 | DIVM__1;           // Set all dividers
    CSCTL0_H = 0;                                   // Lock CS registers
}
// Configure Multiplier Registers
void setMultiplier(void){
```

```
34    MPY32CTL0 = MPYDLYWRTEN | OP1_32 | OP2_32 | MPYM_MACS | MPYSAT_L;
35  }
36  // Configure USCI_A0 for UART mode
37  void setUART(void){
38      UCA0CTLW0 = UCSWRST;                        // Modified only ! 0X0001 (Reset)
39      UCA0CTLW0 |= UCSSEL__SMCLK;                 // SMCLK as Clock source - BRCLK
40      // Baud Rate calculation
41      // 16000000/(16*230400) = 4.34028        // OSR > 16 - Use OSR==16
42      // Fractional portion = 0.34028
43      // User's Guide Table 24-4: UCBRSx = 0x55
44      // UCBRFx = int ( (4.34028-4)*16) = int(5.44)=5
45      UCA0BR0 = 4;                                // 16000000/16/230400
46      UCA0BR1 = 0x00;
47      UCA0MCTLW |= UCOS16 | UCBRF_5 | 0x5500;
48      UCA0CTLW0 &= ~UCSWRST;                      // Initialize eUSCI
49      UCA0IE |= UCRXIE;                           // Enable USCI_A0 RX interrupt
50      __bis_SR_register(GIE);
51      // __bis_SR_register(LPM0_bits | GIE);     // Enter LPM3, interrupts enabled
52  }
53  //-------------------Interrupt Service Routine (ISR)--------------------------//
54  #pragma vector=USCI_A0_VECTOR
55  __interrupt void USCI_A0_ISR(void)
56  {
57    switch(__even_in_range(UCA0IV, USCI_UART_UCTXCPTIFG)){
58    case USCI_NONE: break;
59    case USCI_UART_UCRXIFG:          //interupt based on priority
60      while(!(UCA0IFG & UCTXIFG));
61      //-------------------LOADING INPUT BUFFER-----------------//
62      inputy=UCA0RXBUF;                 //first 8 bits
63      while  (!(UCA0IFG&UCRXIFG));
64      inputy_MSB=UCA0RXBUF;             //second 8 bits
65      inputy+=inputy_MSB<<8;
66      //----------CALLING BASELINE WANDER ESTIMATION----------//
67      processing(inputy, &output);
68      //-------------------LOADING OUTPUT BUFFER----------------//
69      ptr= (unsigned char*) &output;
70      UCA0TXBUF=*ptr++;
71      while  (!(UCA0IFG&UCTXIFG));
72      UCA0TXBUF=*ptr++;
73      while  (!(UCA0IFG&UCTXIFG));
74      UCA0TXBUF=*ptr++;
75      while  (!(UCA0IFG&UCTXIFG));
76      UCA0TXBUF=*ptr;
77        __no_operation();
78        __bic_SR_register_on_exit(LPM0_bits);
79        break;
80      //-----------------------------------------------------//
81      case USCI_UART_UCTXIFG: break;
82      case USCI_UART_UCSTTIFG: break;
83      case USCI_UART_UCTXCPTIFG: break;
```

```
  }
}
//-------------------------------------Main.c-------------------------------------//
int main(void) {
    WDTCTL = WDTPW | WDTHOLD; // Stop watchdog timer
//------------------ Initilasing Ports A,B,C,D,E--------------------//
    P1DIR = 0xFF;
    P1OUT = 0x00;
    P2DIR = 0xFF;
    P2OUT = 0x00;
    P3DIR = 0xFF;
    P3OUT = 0x00;
    P4DIR = 0xFF;
    P4OUT = 0x00;
    P5DIR = 0xFF;
    P5OUT = 0x00;
    P6DIR = 0xFF;
    P6OUT = 0x00;
    P7DIR = 0xFF;
    P7OUT = 0x00;
    P8DIR = 0xFF;
    P8OUT = 0x00;
    P9DIR = 0xFF;
    P9OUT = 0x00;
    P10DIR = 0xFF;
    P10OUT = 0x00;
//--------------------- UART Pin Selection -------------------------//
    P2SEL0 |= BIT0 | BIT1;                  // USCI_A0 UART operation
    P2SEL1 &= ~(BIT0 | BIT1);               // Configure GPIO
//----Disable the GPIO power-on default high-impedance mode----//
    PM5CTL0 &= ~LOCKLPM5;

// FRAM waitstate configuration as required by the device datasheet
// MCLK above 8MHz before the system clock setting.
  FRCTL0 = FRCTLPW | NWAITS_1;
//-----------------------Load Registers-----------------------------//
    setClock();
    setMultiplier();
    setUART();
  return 0;
}
//-----------------Baseline Wander Estimation Function-----------------------//
void processing(float original_inputy, float *pInterpolation_Output){
  static int i=0,counter=0, counter2=0, counter3=0,same_distance_flag=0;
  static float filtered_inputy, moving_integrator_inputy,interpolator_inputx
    [3]={0},interpolator_inputy[3]={0},fiducial_point_inputx=0,
    fiducial_point_inputy=0;
  static float output,processed_signal=0;
  static float filteredinput[12];
  static float unfilteredinput[8];
```

```
132
133   unfilteredinput[counter2]=original_inputy;
134           if (counter==3){
135
136               filter_stage_level(&original_inputy,&output);
137               filtered_inputy=output;
138               filteredinput[counter3]=filtered_inputy;
139
140               Pan_Tompkins(&filtered_inputy,&output);
141               moving_integrator_inputy=output;
142
143               fiducial_point_detect(&unfilteredinput[counter2],&filteredinput[
    counter3],&moving_integrator_inputy,&fiducial_point_inputx,&
    fiducial_point_inputy,counter2,counter3);
144
145               if (interpolator_inputx[2]!=fiducial_point_inputx){
146                   for (i=0;i<2;i++){
147                       interpolator_inputx[i]=interpolator_inputx[i+1];
148                       interpolator_inputy[i]=interpolator_inputy[i+1];
149                       }
150                   interpolator_inputx[2]=fiducial_point_inputx;
151                   interpolator_inputy[2]=fiducial_point_inputy;
152                   if (interpolator_inputx[2]-interpolator_inputx[1]==
    interpolator_inputx[1]-interpolator_inputx[0]){
153                       same_distance_flag=1;
154                   }
155               }
156               else {
157               interpolator_inputx[2]=fiducial_point_inputx;
158               interpolator_inputy[2]=fiducial_point_inputy;
159               }
160               counter=0;
161               counter3++;
162           }
163           counter++;
164           counter2++;
165           if (counter2==8)
166               counter2=0;
167           if (counter3==12)
168               counter3=0;
169           if (interpolator_inputx[0]>0){
170               hybrid_interpolation(&interpolator_inputx[2], &interpolator_inputy
    [2],&processed_signal,&same_distance_flag);
171           }
172           *pInterpolation_Output=processed_signal;
173   }
```

Appendix F

MATLAB Code: Additional Algorithm

```
%% FILTER DESIGN BLOCK
%
% This block designs the high-pass and low-pass filter characteristics and
% returns coefficient parameters to the main function

function [LP_coeff_numerator,LP_coeff_denominator,HP_coeff_numerator,
    HP_coeff_denominator] = bandpass_filter_design_block(filter_type,Fp_lp,Fst_lp,
    Ap_lp,Ast_lp,Fst_hp,Fp_hp,Ast_hp,Ap_hp)

%-------------------------------LOW-PASS FILTER DESIGN-------------------------------

d=fdesign.lowpass('Fp,Fst,Ap,Ast',Fp_lp,Fst_lp,Ap_lp,Ast_lp); %low-pass design
temp=designmethods(d);                                        %design methods
if strcmp(temp(filter_type),'equiripple')==1
    Hd_Low_Pass = design(d,temp{filter_type});
    fvtool(Hd_Low_Pass); %filter_responses
else
    Hd_Low_Pass = design(d,temp{filter_type},'matchexactly','stopband');
    fvtool(Hd_Low_Pass); %filter_responses
end
% butter=1, cheby1=2, cheby2=3, ellip=4, equiripple=5, ifir=6, kaiserwin=7
if strcmp(temp(filter_type),'equiripple')==1
    LP_coeff_numerator=Hd_Low_Pass.numerator;
    LP_coeff_denominator=1;
elseif strcmp(temp(filter_type),'butter')==1
    [LP_coeff_numerator,LP_coeff_denominator] = sos2tf(Hd_Low_Pass.sosMatrix,
    Hd_Low_Pass.ScaleValues);
elseif strcmp(temp(filter_type),'cheby1')==1
    [LP_coeff_numerator,LP_coeff_denominator] = sos2tf(Hd_Low_Pass.sosMatrix,
    Hd_Low_Pass.ScaleValues);
elseif strcmp(temp(filter_type),'cheby2')==1
    [LP_coeff_numerator,LP_coeff_denominator] = sos2tf(Hd_Low_Pass.sosMatrix,
    Hd_Low_Pass.ScaleValues);
elseif strcmp(temp(filter_type),'ellip')==1
```

```
    [LP_coeff_numerator,LP_coeff_denominator] = sos2tf(Hd_Low_Pass.sosMatrix,
    Hd_Low_Pass.ScaleValues);
elseif strcmp(temp(filter_type),'ifir')==1
    [LP_coeff_numerator,LP_coeff_denominator] = sos2tf(Hd_Low_Pass.sosMatrix,
    Hd_Low_Pass.ScaleValues);
elseif strcmp(temp(filter_type),'kaiserwin')==1
    LP_coeff_numerator=Ld_Low_Pass.numerator;
    LP_coeff_denominator=1;
end

%-------------------------------HIGH-PASS FILTER DESIGN-------------------------------

d=fdesign.highpass('Fst,Fp,Ast,Ap',Fst_hp,Fp_hp,Ast_hp,Ap_hp); %high-pass design
temp=designmethods(d);                                          %design methods
if strcmp(temp(filter_type),'equiripple')==1
    Hd_High_Pass = design(d,temp{filter_type});
    fvtool(Hd_High_Pass); %filter_responses
else
    Hd_High_Pass = design(d,temp{filter_type},'matchexactly','passband');
    fvtool(Hd_High_Pass); %filter_responses
end
% butter=1, cheby1=2, cheby2=3, ellip=4, equiripple=5, ifir=6, kaiserwin=7
if strcmp(temp(filter_type),'equiripple')==1
    HP_coeff_numerator=Hd_High_Pass.numerator;
    HP_coeff_denominator=1;
elseif strcmp(temp(filter_type),'butter')==1
    [HP_coeff_numerator,HP_coeff_denominator] = sos2tf(Hd_High_Pass.sosMatrix,
    Hd_High_Pass.ScaleValues);
elseif strcmp(temp(filter_type),'cheby1')==1
    [HP_coeff_numerator,HP_coeff_denominator] = sos2tf(Hd_High_Pass.sosMatrix,
    Hd_High_Pass.ScaleValues);
elseif strcmp(temp(filter_type),'cheby2')==1
    [HP_coeff_numerator,HP_coeff_denominator] = sos2tf(Hd_High_Pass.sosMatrix,
    Hd_High_Pass.ScaleValues);
elseif strcmp(temp(filter_type),'ellip')==1
    [HP_coeff_numerator,HP_coeff_denominator] = sos2tf(Hd_High_Pass.sosMatrix,
    Hd_High_Pass.ScaleValues);
elseif strcmp(temp(filter_type),'ifir')==1
    [HP_coeff_numerator,HP_coeff_denominator] = sos2tf(Hd_High_Pass.sosMatrix,
    Hd_High_Pass.ScaleValues);
elseif strcmp(temp(filter_type),'kaiserwin')==1
    HP_coeff_numerator=Hd_High_Pass.numerator;
    HP_coeff_denominator=1;
end
if length(temp)~=7
    error('Wrong Filter Type')
end
end
```

```
%% PT INTERVAL ERROR BLOCK
%
% This block designs calculates the RMS PT interval errors defined
% by the baseline wander estimation algorithm

function [detected_error] = PT_interval_error(savefile,file)

%-------------------------LOAD FILES & READ ANNOTATIONS-------------------------

for i=1:length(file)
  ann=rdann(strcat('mcode/MIT_BIH_Arrythmia_Annotations/',file(1:3)),'atr'
    ,[],[],1);
    load(savefile);
    for i=5:length(ann)-30 % Discard final 30 samples if FIR filter is used
    T=find(locationofzeros>ann(i-1) & locationofzeros<ann(i));
    P=find(locationofzeros>ann(i-2) & locationofzeros<ann(i-1));
        if length(T)<2 || isempty(P)
        detected_error(i)=rms(difference(ann(i-1):ann(i)));
        else
        T_s=sort(T);
        T_s_p=T_s(2);
        P_s=max(P);
        detected_error(i)=rms(difference(locationofzeros(P_s)-50:locationofzeros(
    T_s_p)));
        end
    end
    detected_error=detected_error(16:end); % discregarding the initalization
    error
    detected_error=detected_error*1000/200; % use microvolt error (Recording has
    200V/V as gain)

%-----------------WITH MOTION ARTEFACT MEAN MEDIAN STD-----------------

    MEAN_err=mean(detected_error);
    MEDIAN_err=median(detected_error);
    STD_err=std(detected_error);

%-----------------WITHOUT MOTION ARTEFACT MEAN MEDIAN STD-----------------

    WMA_MEAN_err=mean(detected_error(find(detected_error<200)));
    WMA_MEDIAN_err=median(detected_error(find(detected_error<200)));
    WMA_STD_err=std(detected_error(find(detected_error<200)));

%-----------------------------------FILE SAVE-----------------------------------
    save(savefile, 'detected_error','MEAN_err','MEDIAN_err','STD_err','
    WMA_MEAN_err','WMA_MEDIAN_err','WMA_STD_err','-append');

end
```

```
%% SYNTHETIC DATA SEGMENT ANALYSIS
% Synthetic data generated through Karthik's code is segmented with the
% code defined covered here. Plot functions are not included due to space
% requirements

function [P_onset,P_offset,Q_onset,S_offset,T_onset,T_offset] = segmentanalysis(
    joinedECG)

%-----------------------------------SEGMENTATION CALCULATIONS-----------------------------------

val=joinedECG;
g_val=gradient(val);
square_g_val=g_val.^2;
second_g_val=gradient(g_val);
square_second_g_val=second_g_val.^2;
%Defining a moving average filter to suppress EMG if exists
a=1;
b=[0.25 0.25 0.25 0.25];
square_second_g_val=filter(b,a,square_second_g_val);
%Synthetic threshold
threshold=zeros(1, length(val));
threshold=threshold+0.8;
detected_threshold=find(square_second_g_val<threshold.^2);

%-----------------------------------FIND FIDUCIAL LOCATIONS-----------------------------------

j=1;
for i=1:length(detected_threshold)-1
    if detected_threshold(i)+1 ~= detected_threshold(i+1)
        [r]=find(second_g_val(detected_threshold(i):detected_threshold(i+1))>max(
    second_g_val((detected_threshold(i):detected_threshold(i+1)))-0.0001));
        if length(r)>1
            r=r(1);
        end
        locationoffiducial(j)=detected_threshold(i)+r-1;
        j=j+1;
    end
end
if ~exist('locationoffiducial','var')
    error('Lower your segment detection threshold value')
end

%-----------------------------------DEFINE FIDUCIAL POINTS-----------------------------------

P_onset = locationoffiducial(1);
P_offset = locationoffiducial(2);
Q_onset = locationoffiducial(3);
S_offset = locationoffiducial(7);
T_onset  = locationoffiducial(8);
T_offset = locationoffiducial(9);
```

```
49
50 %-------------------------------MATRIX FORMATION-------------------------------
51
52 for j=1:floor((length(locationoffiducial))/9)-1
53
54     P_onset_temp=locationoffiducial(1+j*9);
55     P_onset=[P_onset P_onset_temp];
56
57     P_offset_temp=locationoffiducial(2+j*9);
58     P_offset=[P_offset P_offset_temp];
59
60     Q_onset_temp=locationoffiducial(3+j*9);
61     Q_onset=[Q_onset Q_onset_temp];
62
63     S_offset_temp=locationoffiducial(7+j*9);
64     S_offset=[S_offset S_offset_temp];
65
66     T_onset_temp=locationoffiducial(8+j*9);
67     T_onset=[T_onset T_onset_temp];
68
69     T_offset_temp=locationoffiducial(9+j*9);
70     T_offset=[T_offset T_offset_temp];
71
72 end
73 end
74
75 %-------------------------------------------------------------------------------
76 %-------------------------------------------------------------------------------
77
78
79 %% HISTOGRAM FUNCTION UTILISING SYNTHETIC SEGMENT ANALYSIS
80 %
81 % This block uses the detected segment locations by the segment analysis
82 % function to find RMS errors within these segments
83
84 function [DB_e,DB_e_ST,DB_e_PRi,DB_e_PRs,DB_e_QRSc,DB_e_QTi] = histogram(
       detected_error,locationofzeros,P_onset,P_offset,Q_onset,S_offset,T_onset,
       T_offset)
85
86 %-------------------------------ERROR IN MICRO VOLTS-------------------------------
87
88 %detected_error=detected_error(1500:end); % discregarding the initalization error
89 detected_error=detected_error*1000/200;   % microvolt error
90
91 x=[P_onset(1)-locationofzeros(1) P_offset(1)-locationofzeros(1) Q_onset(1)-
       locationofzeros(1) S_offset(1)-locationofzeros(1) T_onset(1)-locationofzeros
       (1) T_offset(1)-locationofzeros(1)];
92 [r]=find(x<0);
93 r=max(r)+1;
94
```

```
%-------------------------SYNTHETIC DATA FIDUCIAL POINT-------------------------

if r==2
    P_onset(1)=locationofzeros(1);
elseif r==3
    P_onset(1)=locationofzeros(1);
    P_offset(1)=locationofzeros(1);
elseif r==4
    P_onset(1)=locationofzeros(1);
    P_offset(1)=locationofzeros(1);
    Q_onset(1)=locationofzeros(1);
elseif r==5
    P_onset(1)=locationofzeros(1);
    P_offset(1)=locationofzeros(1);
    Q_onset(1)=locationofzeros(1);
    S_offset(1)=locationofzeros(1);
else
    error('Reset T_onset & T_offset values')
end

%-------------------------------DEFINING INTERVALS------------------------------

PR_interval_max=(Q_onset-P_onset)+1;
PR_segment_max=(Q_onset-P_offset)+1;
QRS_complex_max=(S_offset-Q_onset)+1;
QT_interval_max=(T_offset-Q_onset)+1;
ST_segment_max=(T_onset-S_offset)+1;
for i=1:length(P_onset)
    PR_interval(i,1:PR_interval_max(i))=P_onset(i):Q_onset(i);
    PR_segment(i,1:PR_segment_max(i))=P_offset(i):Q_onset(i);
    QRS_complex(i,1:QRS_complex_max(i))=Q_onset(i):S_offset(i);
    QT_interval(i,1:QT_interval_max(i))=Q_onset(i):T_offset(i);
    ST_segment(i,1:ST_segment_max(i))=S_offset(i):T_onset(i);
end
PR_interval_hist=0;
PR_segment_hist=0;
QRS_complex_hist=0;
QT_interval_hist=0;
ST_segment_hist=0;
sizePR_int=size(PR_interval);
for j=1:sizePR_int(1)
PR_interval_hist= [PR_interval_hist PR_interval(j,:)];
PR_segment_hist= [PR_segment_hist PR_segment(j,:)];
QRS_complex_hist= [QRS_complex_hist QRS_complex(j,:)];
QT_interval_hist= [QT_interval_hist QT_interval(j,:)];
ST_segment_hist= [ST_segment_hist ST_segment(j,:)];
end
PR_interval_hist(:,PR_interval_hist==0)=[];
PR_segment_hist(:,PR_segment_hist==0)=[];
QRS_complex_hist(:,QRS_complex_hist==0)=[];
```

```
145 QT_interval_hist(:,QT_interval_hist==0)=[];
146 ST_segment_hist(:,ST_segment_hist==0)=[];
147
148 ST_segment_hist=ST_segment_hist.*(ST_segment_hist<locationofzeros(1)+
    locationofzeros(length(locationofzeros)));
149 ST_segment_hist(:,ST_segment_hist==0)=[];
150 ST_segment_hist=ST_segment_hist(1:find(ST_segment_hist<length(detected_error),1,'
    last'));
151 PR_interval_hist=PR_interval_hist.*(PR_interval_hist<locationofzeros(1)+
    locationofzeros(length(locationofzeros)));
152 PR_interval_hist(:,PR_interval_hist==0)=[];
153 PR_interval_hist=PR_interval_hist(1:find(PR_interval_hist<length(detected_error)
    ,1,'last'));
154 PR_segment_hist=PR_segment_hist.*(PR_segment_hist<locationofzeros(1)+
    locationofzeros(length(locationofzeros)));
155 PR_segment_hist(:,PR_segment_hist==0)=[];
156 PR_segment_hist=PR_segment_hist(1:find(PR_segment_hist<length(detected_error),1,'
    last'));
157 QRS_complex_hist=QRS_complex_hist.*(QRS_complex_hist<locationofzeros(1)+
    locationofzeros(length(locationofzeros)));
158 QRS_complex_hist(:,QRS_complex_hist==0)=[];
159 QRS_complex_hist=QRS_complex_hist(1:find(QRS_complex_hist<length(detected_error)
    ,1,'last'));
160 QT_interval_hist=QT_interval_hist.*(QT_interval_hist<locationofzeros(1)+
    locationofzeros(length(locationofzeros)));
161 QT_interval_hist(:,QT_interval_hist==0)=[];
162 QT_interval_hist=QT_interval_hist(1:find(QT_interval_hist<length(detected_error)
    ,1,'last'));
163
164 %-------------------------INTERVAL ERROR ANALYSIS-------------------------------
165 %-----------------------RMS error R-R-----------------------
166
167 j=1;
168 for i=1:floor((find(locationofzeros<length(detected_error), 1, 'last')-1)/3)
169         DB_e(j)=rms(detected_error(locationofzeros(3*i-2):locationofzeros(3*i+1))
    );
170         j=j+1;
171 end
172 if mod(length(locationofzeros)-1,3)~=0
173         DB_e(j)=rms(detected_error(locationofzeros(3*i+1:find(locationofzeros<
    length(detected_error), 1, 'last'))));
174 end
175
176 %-------------------RMS error ST segment-------------------
177 j=1;
178 temp=0;
179 for i=1:length(ST_segment_hist)-1
180     if ST_segment_hist(i+1)~=ST_segment_hist(i)+1
181         DB_e_ST(j)=rms(detected_error(ST_segment_hist(temp+1:i)));
182         temp=i;
```

```
183             j=j+1;
184         end
185     end
186
187     % ----------------------RMS error PR interval----------------------
188     j=1;
189     temp=0;
190     for i=1:length(PR_interval_hist)-1
191         if PR_interval_hist(i+1)~=PR_interval_hist(i)+1
192             DB_e_PRi(j)=rms(detected_error(PR_interval_hist(temp+1:i)));
193             temp=i;
194             j=j+1;
195         end
196     end
197
198     % ----------------------RMS error PR segment----------------------
199     j=1;
200     temp=0;
201     for i=1:length(PR_segment_hist)-1
202         if PR_segment_hist(i+1)~=PR_segment_hist(i)+1
203             DB_e_PRs(j)=rms(detected_error(PR_segment_hist(temp+1:i)));
204             temp=i;
205             j=j+1;
206         end
207     end
208
209     % ----------------------RMS error QRS complex----------------------
210     j=1;
211     temp=0;
212     for i=1:length(QRS_complex_hist)-1
213         if QRS_complex_hist(i+1)~=QRS_complex_hist(i)+1
214             DB_e_QRSc(j)=rms(detected_error(QRS_complex_hist(temp+1:i)));
215             temp=i;
216             j=j+1;
217         end
218     end
219
220     % ----------------------RMS error QT interval----------------------
221     j=1;
222     temp=0;
223     for i=1:length(QT_interval_hist)-1
224         if QT_interval_hist(i+1)~=QT_interval_hist(i)+1
225             DB_e_QTi(j)=rms(detected_error(QT_interval_hist(temp+1:i)));
226             temp=i;
227             j=j+1;
228         end
229     end
230     end
```