Data reduction algorithm for on-line ECG applications

By
Azita Dordari

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfilment of
the requirements for the degree of

Master of Computer Science

Ottawa-Carleton Institute for Computer Science
School of Computer Science
Carleton University
Ottawa, Ontario

January 2006

Your file Votre référence
ISBN: 978-0-494-16481-5
Our file Notre référence
ISBN: 978-0-494-16481-5

Abstract

Signal compression is an important problem encountered in many applications. Various techniques have been proposed over the years for addressing the problem. The main focus of this thesis is to present an algorithm for compressing digital Electrocardiogram (ECG) signals in on-line applications with a continuous stream of data. Two main compression algorithms were developed:

- Least Area algorithm. This algorithm processes the signal in on-line applications. It divides the signal into sub signals and selects a subset of it by minimizing the area between the original and compressed set within a certain threshold. This approach guarantees a high quality signal in linear time.
- Some modifications have been done on the CCSP algorithm which has very good compressed signals. A new error definition based on minimizing the area between original and compressed set enabled the algorithm to achieve a low error within a certain compression ratio. In addition to the ability of processing a continuous signal which CCSP does not have, the running time of the algorithm has also been improved.

Using a varied signal test set, extensive coding experiments are presented. Results from these two methods are compared with traditional time domain ECG compression methods, as well as, with more recently developed frequency domain methods. Evaluation is based on the percentage root mean square difference (PRD) performance measure, the maximum error, execution time and visual inspection of the reconstructed signals. The results demonstrate that LA has excellent performance compared to traditional ECG compression methods and can be used with on-line applications. The experiments have also shown that MCCSP not only has been improved in terms of running time, but it also generates a more detailed signal.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The Electrocardiogram (ECG) is a recording (measurement) of the electrical activity of the heart. The ECG signal is captured as an analog signal. In recent years the development of digital technology allows the signal to be also converted to digital signals. Analysis of the signal provides the most common non-invasive method for diagnosis of Cardiac dysfunctions. Different diseases require different types of signals to be captured.

The term *signal* often refers to a continuous time and amplitude signal. We are concerned with digital signals, which means the signal is obtained by sampling and quantizing the continuous signal. Therefore in this thesis we use the word signal interchangeably with digital signal. In cases where we refer to a sequence of numbers, or vectors, the word *data* is often used as a synonym for signal. *Compression* is referred to signal compression. In this context we use the term *signal compression* and by that we understand a process intended to provide efficient representation of the signal while preserving the essential information contained within it. When talking about a compression technique or a compression algorithm we are actually referring to two sub algorithms: A compression algorithm accepts as *input* a signal Y and generates a set C of compressed *output indices set.* A reconstruction algorithm then processes the compressed index representation C to generate the *reconstructed set* $\hat{Y}$.

A typical ECG monitoring device generates large amounts of digital data. The sampling rate typically varies from 125 to 500 Hz and each data sample may have 8, 12 or 16-bit resolution. This leads to a large amount of ECG data from 60 Kbits per minute to 480 Kbits per minute [27].

Some of the applications in which the ECG compression is clearly necessary are as follows:

- ambulatory ECG for the detection of heart block transients.
- real-time patient monitoring in coronary care units and intensive care units.
- on-line real-time patient monitoring.

Since large amount of data is collected at an ECG recording for example from different sensors placed on the patient's body and sometimes the data should be stored for long time, signal compression is essential for increasing the storage and transmission efficiency and decreasing the cost of storage and transmission.

In on-line monitoring, it is required to transfer the ECG data from a patient to a health center. Normally, the device attached to the patient's body is small; therefore, minimum hardware resources including CPU, RAM, and the storage are used. On the other side, the amount of the data should be small. Finally, the result should be accurate in a way that a physician is able to capture the important ECG components visually. As a result, there should be a simple technique which is able to execute fast and needed small hardware resources to compress the ECG signal.

Compression algorithms can be classified as either *lossless*, which involve no loss of information, or *lossy* where the signal generally cannot be recovered or reconstructed exactly. The algorithms we focus on, belong to lossy algorithms. To obtain a significant degree of compression, a small amount of distortion can be tolerated. Exactly what is a "small amount" of distortion depends on the particular application.

Given an *Area threshold*, we aim to find a high quality compressed signal. The

problem of finding a sub-sequence of the original signal in a way that the area between these two sequences is less than a specified *Area threshold*, is called a Least Area problem. The solution is based on a mathematical foundation and is one possible solution under the given constraints. We approximate the signal by extracting information from it. The extracted information leads to a compressed representation of the original signal.

Two different algorithms have been developed in this thesis:

1- Least Area is a new on-line algorithm which takes an acceptable error as input and generates a good quality compressed graph and can perform in linear time.

2- MCCSP is a modification of the Cardinality Constraint Shortest Path (CCSP). CCSP is a high quality signal compression algorithm, which is not suitable for on-line applications. The modified algorithm uses the compression ratio as an input to generate a good quality graph within less time than the original algorithm. The *Error* depends on the compression ratio given to the algorithm.

This thesis is organized as follows: Chapter 2 describes the ECG signal and signal analysis. Background information of the ECG data compression techniques is also presented in this chapter. Chapter 3 presents the Least Area algorithm in details. Chapter 4 discusses the modified CCSP algorithm. Chapter 5 includes the experimental results. Chapter 6 summarizes the thesis.

Chapter 2

Literature Review

This chapter provides some background information about ECG. The first and second sections are devoted to the components of an ECG signal and leads to a brief overview of signal analysis. Third section reviews some major ECG compression algorithms and their strength and weaknesses.

2.1 The Electrocardiogram

2.1.1 Electrical Basis of the Electrocardiogram

An electrocardiogram (ECG) is a graphical record of the changes in magnitude and direction of the electrical activity, or more specifically, the electric current that is generated by the depolarization and repolarization of the atria and ventricles [12]. This electrical activity is readily detected by electrodes attached to the skin. But neither the electrical activity that results from the generation and transmission of electrical impulses, which are too feeble to be detected by skin electrodes, nor the mechanical contractions and relaxations of the atria and ventricles (which do not generate electrical activity) appear in the electrocardiogram.

2.1.2 Components of the Electrocardiogram

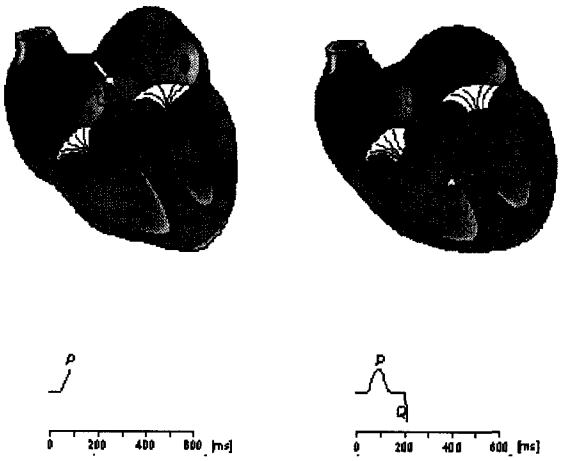
The electric current which is generated by depolarization and repolarization of the atria and ventricles, is detected by electrodes which are attached to the patient's body. The current is then amplified and displayed on an oscilloscope, recorded on ECG paper or stored in memory [18].

The electric current which is generated by atrial depolarization is recorded as the P wave and the current which is generated by ventricular depolarization, is recorded as Q, R, and S waves: the QRS complex. Atrial repolarization is recorded as the atrial T wave and ventricular repolarization, as the ventricular T wave or simply, the T wave. Since atrial repolarization normally occurs during ventricular depolarization, the atrial T wave is buried or hidden in the QRS complex. Figure 2.1 and 2.2 show the relation between the ECG graph and heart activities. After the electric activation of the heart has begun, it spreads along the atrial walls. The resultant vector of the atrial electric activity is illustrated with a thick arrow. After the depolarization has propagated over the atrial walls, it makes the P wave. QRS complex is generated by Septal,Apical and Ventricular depolarization. T wave is generated by ventricular depolarization.

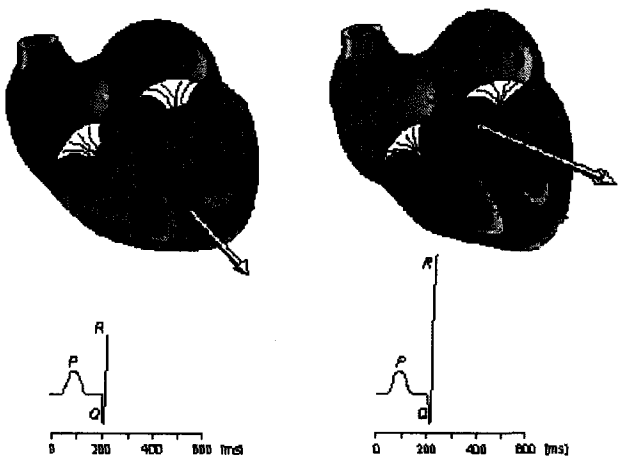
In a normal cardiac cycle, the P wave occurs first, followed by the QRS complex and the T wave (Figure 2.3).

The sections of the ECG between the waves and complexes are called segments and intervals:

- P wave: the sequential activation (depolarization) of the right and left atri
- PR segment: the portion on the ECG wave from the end of the P wave to the beginning of the QRS complex
- QRS complex: right and left ventricular depolarization (normally the ventricles are activated simultaneously)
- ST-T wave: ventricular repolarization

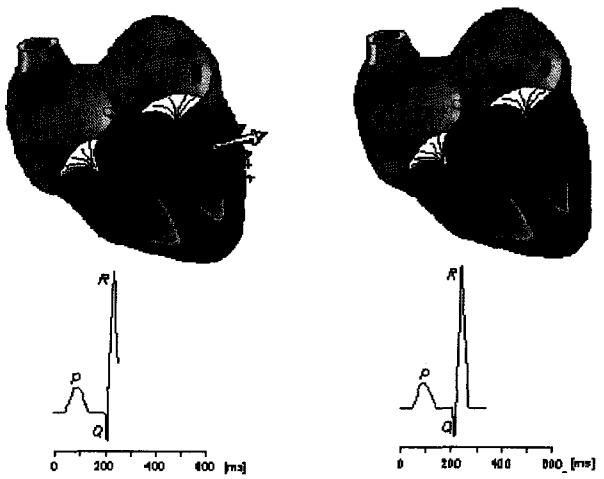


(a) Atrial Depolarization and Septal Depolarization

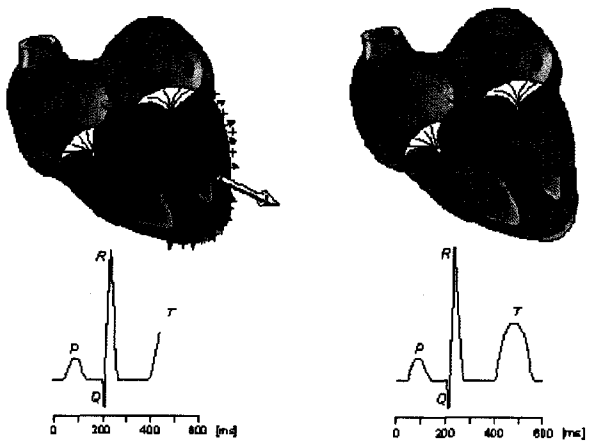


(b) Apical Depolarization and Left Ventricular Depolarization

Figure 2.1: Recording ECG- part 1 [18]



(a) Late Left Ventricular Depolarization and Ventricles Depolarized



(b) Ventricular Repolarization and Ventricles Repolarized

Figure 2.2: Recording ECG- part 2 [18]

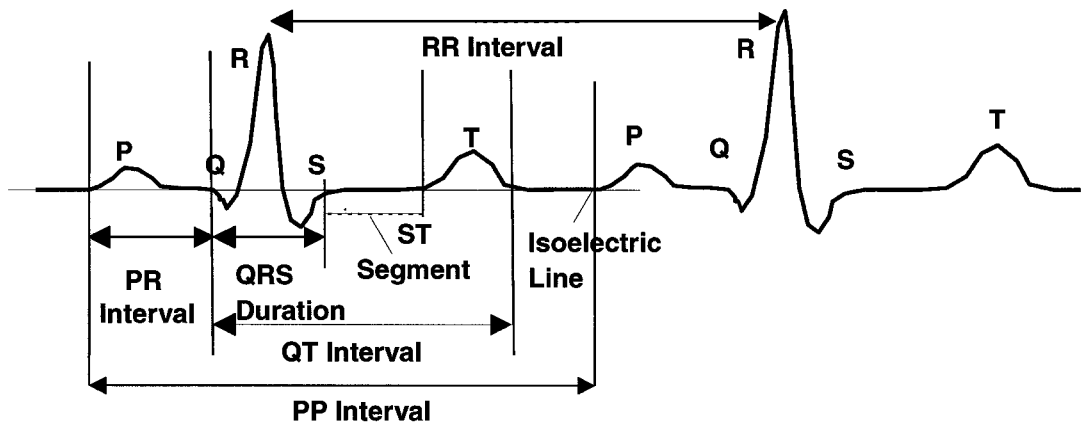


Figure 2.3: Normal cardiac cycle

- U wave: origin for this wave is not clear - but probably represents what has happened after depolarizations in the ventricles
- PR interval: time interval from onset of atrial depolarization (P wave) to onset of ventricular depolarization (QRS complex)
- QRS duration: duration of ventricular muscle depolarization
- ST segment: segment between the end of the QRS complex and the initial deflection of the T-wave
- QT interval: duration of ventricular depolarization and repolarization
- RR interval: duration of ventricular cardiac cycle (an indicator of ventricular rate)
- PP interval: duration of atrial cycle (an indicator or atrial rate)

2.1.3 ECG Leads

An ECG lead is a record (spatial sampling) of the electrical activity generated by the heart that is sensed by one of these two methods: (1) two discrete electrodes of opposite polarity or (2) one discrete positive electrode and an indifferent zero reference point. A lead composed of two discrete electrodes of opposite polarity is called a bipolar lead and a lead composed of a single discrete positive electrode and a zero reference point is a unipolar lead [15].

Depending on the ECG lead being recorded, the positive electrode may be attached to the right or left arm, the left leg or one of several locations on the anterior chest wall. The negative electrode is usually attached to an opposite arm or leg or to a reference point made by connecting the limb electrodes together.

A detailed analysis of the heart's electrical activity is critical in a hospital setting using ECG signal from 12 separate leads (the 12-lead ECG). The 12-lead ECG is also used in the prehospital phase of emergency care in certain advanced life support services to diagnose acute myocardial infraction and to help in the identification of certain arrhythmias.

A 12-lead ECG consists of three standard (bipolar) limb leads (I, II, and III) in Figure 2.4, three augmented (unipolar) leads (aV_R, aV_L, and aV_F) (Figure 2.5), and six precordial (unipolar) leads (V_1, V_2, V_3, V_4, V_5, and V_6) (Figure 2.6) [15].

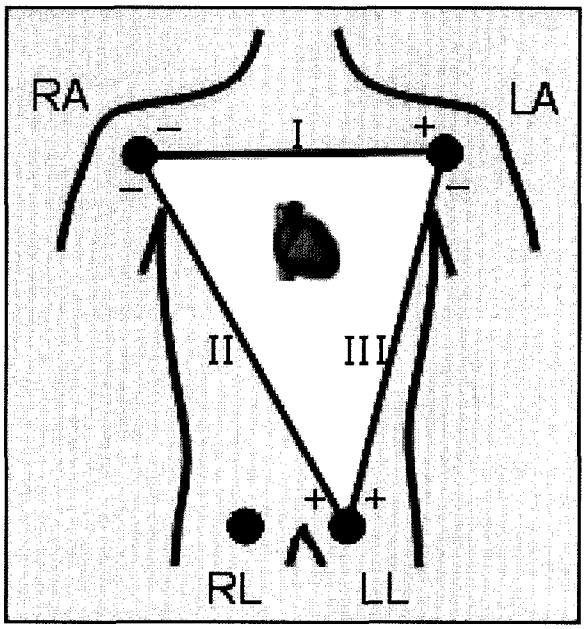


Figure 2.4: The standard (bipolar) limb leads I, II, and III.

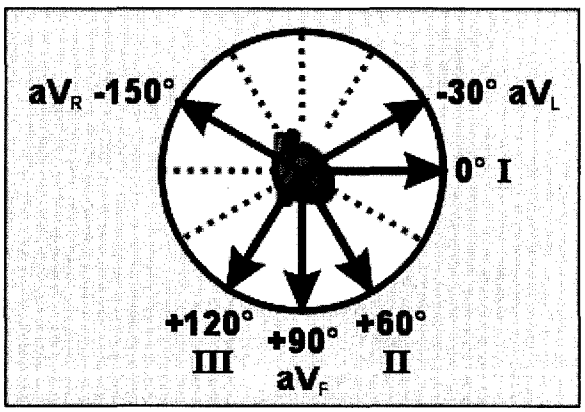


Figure 2.5: The augmented (unipolar) leads aV_R, aV_L, and aV_F.

2.2 Signal Analysis

Diagnosis of the normal electrocardiogram is made by excluding any recognized abnormality. A short summary of some of the normalities and abnormalities of the ECG element are discussed next [14, 26].

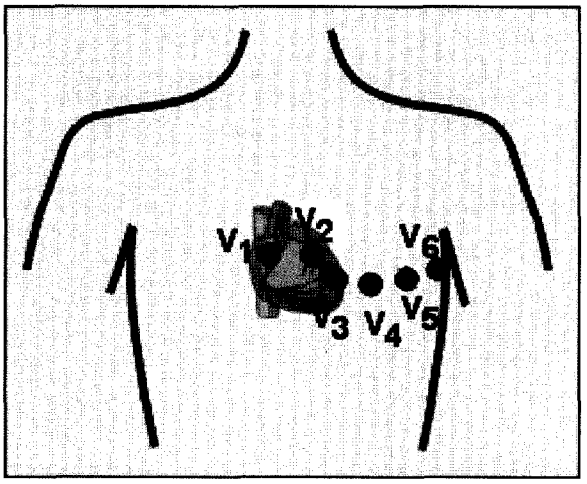


Figure 2.6: Precordial (unipolar) leads

The horizontal axis on an ECG paper represents time. The paper contains a grid of small blocks which are grouped together to form larger blocks. Large blocks are 0.2 seconds in duration, while small blocks are 0.04 seconds in duration. The vertical axis represents voltage. Large blocks are 5mm, while small blocks represent 1mm (Figure 2.7). Every component of the ECG has an specific characteristic, for some of them the exact number in height or width is important while in others the slope of the segment represents a disease. The diagnosis of the normal electrocardiogram is made by excluding any recognized abnormality by a physician. It can also provide clues about other heart conditions and certain medical conditions. The ECG frequency differs from 150Hz to 500Hz which means the time interval between two samples can vary between 0.001 seconds and 0.006 seconds.

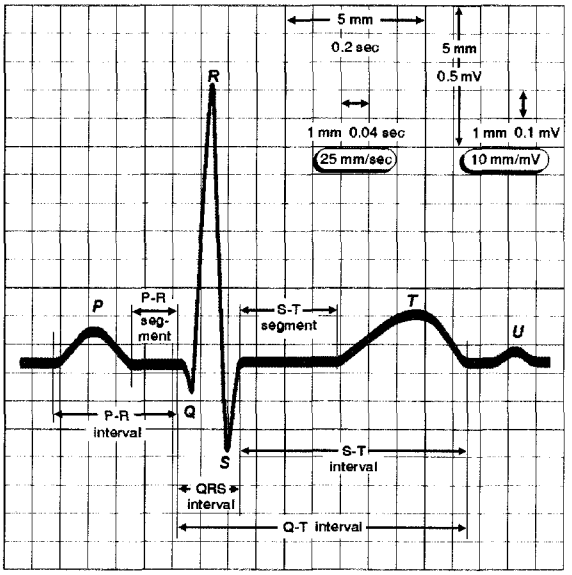


Figure 2.7: ECG Paper

In a normal *sinus rhythm*, each P wave is followed by a QRS, and is usually 60 to 100 beats per minute in duration.

If the *P rate* is less than 60 it indicates *sinus bradycardia* and if it is more than 100 or its variation is more than 10% it shows *sinus tachycardia.*

Normal *P waves* are less than 2.5mm in height and less than 0.11 seconds wide in lead II . Abnormal P waves may indicate right atrial hypertrophy, left atrial hypertrophy, atrial premature beat and hyperkalaemia.

PR interval is normal if it is between 0.12 to 0.20 seconds. A short PR segment is related to Wolff-Parkinson-White syndrome or Lown-Ganong-Levine syndrome, Duchenne muscular dystrophy, type II glycogen storage disease (Pompe's) and HOCM. A long PR interval is a sign of first degree heart block and trifasicular block.

The duration of a normal *QRS complex* is less than 0.12 seconds. If it is wider, it shows right or left bundle branch block, ventricular rhythm, hyperkalaemia.

The value of the *QT interval* is not used in diagnosis by itself. The corrected QT interval (QTc) is calculated by dividing the QT interval by the square root of the preceding R-R interval. Normal corrected QTc intervals are less than 0.44 seconds. Causes of a long QT interval are infarction, myocarditis, diffuse myocardial disease hypocalcaemia, hypothyroidism subarachnoid haemorrhage, intracerebral haemorrhage drugs like sotalol and amiodarone, hereditary Romano Ward syndrome (autosomal dominant) Jervill and Lange Nielson syndrome (autosomal recessive) associated with sensorineural deafness.

A normal *ST segment* has no elevation or depression. Causes of elevation include acute MI like anterior and inferior, left bundle branch block, acute pericarditis, normal variants such athletic heart, Edeiken pattern and high-take off. Depression may be a result of myocardial ischaemia, digoxin effect, ventricular hypertrophy, acute posterior MI, pulmonary embolus and left bundle branch block.

The *T wave* should not be tall or small. If it is tall, hyperkalemia, hypercube myocardial infarction and left bundle branch block may be the diagnosis. Causes of small, flattened or inverted T waves are numerous and include ischaemia, age, race, hyperventilation, anxiety, drinking iced water, LVH, drugs, pericarditis, PE, intraventricular conduction delay and electrolyte disturbance.

The *U wave* should be asymmetric. Prominent upright U waves, negative or inverted may indicate myocardial disease, left main or left anterior descending artery disease.

2.3 ECG Compression Algorithms

Many ECG waveform compression algorithms have been proposed. They can be classified into two main categories:

1. **Time domain lossy signal compression algorithms:**

   These algorithms select a subset of significant signal samples to represent the

original signal [13]. Selection of these significant samples depends on the underlying criterion for the sample selection process. Sample selection criteria is very important because it directly affects the quality of the reconstructed sequence. The original signal is reconstructed by an inverse process, most often by drawing straight lines between the extracted samples. Algorithms in this category include Fan [4], Aztec [3], COTES [2], TP [22] and CCSP [23]. Some of these algorithms contain a preprocessing stage and after selecting a subset of the original sample, the actual compression method is applied such as AZTDIS [30] and TRIM [21, 20]. All the traditional ECG lossy algorithms are based on heuristics in the sample selection process. This generally makes them fast, but they all suffer from sub-optimality.

2. **Lossless ECG compression algorithms:**

These algorithms use a transformation technique to compress the ECG signal. One main category of lossless ECG compression algorithms is the transform compression methods. These methods mainly utilize the spectral and energy distributions of the signal. Generally the input signal is processed by means of some transformation. Signal reconstruction is achieved by an inverse transformation process. This category includes traditional transform coding techniques such as KarhunenLoeve Transform [24], Fourier Transform [25], Cosine Transform and Walsh Transform [16], as well as, subband techniques [1]. The vector Quantization and Wavelet Transform [5]. Although Lossless algorithms preserve most of the characteristic of the signal, they have very low compression ratio. For example, ZIP algorithm can achieve a compression ratio of 0.59 [7]. There are other parameters to be considered when using a lossless algorithm such as the compression quality , the coder delay and the processing requirements and their cost impact. The most efficient voice coders use more network resources for higher data rate [28]. The CELP (Codebook Excited Linear Predictive coding) voice channel modules bring superior quality voice plus an up to 10% compression, but it can not work with the minimum hardware resources which is attached to the patient's body.

Since a fast algorithm with low compression ratio and good quality is very important in on-line applications, we just review the time domain signal compression algorithms. Lossless algorithms can not lower the compression ratio as much as lossy algorithms can. Moreover some of the lossless algorithms are sensitive to the frame size (delay time). In a normal ECG graph, finding the pattern of changes is very easy, but for a patient who needs on-line monitoring, this pattern may change while monitoring. Therefore, lossless algorithms are not suitable for this thesis purpose.

2.3.1 The Fan algorithm

The Fan algorithm is based on first-order interpolation. The algorithm was originally reported and tested on ECG signals by Gardenhire [8]. It has been exhaustively used and tested on ECG signals [4].

This algorithm receives a sequence of signal values $\{y_0, y_1, ..., y_n\}$ as input and selects a subset of them as output. Suppose P_i denotes the point (x_i, y_i). x_i represents the time and y_i is the amplitude value of the point. The first sample point is always selected. ε is the voltage threshold. Next two lines are drawn; One line, $U1$, between P_0 and $(x_1, y_1 + \varepsilon)$ and another line, $L1$, between P_0 and $(x_1, y_1 - \varepsilon)$ as shown in Figure 2.8. If the third sample, P_2, is within the area bounded by the two lines, new slopes (U2;L2) are drawn between the initial point and the third sample point plus the same specified threshold. These new lines (U2; L2) are then compared to the previously stored lines (U1; L1) and the most restrictive lines are retained (U1; L2). The process is repeated, comparing future sample values to the most restrictive lines. Whenever a sample falls outside the area bounded by the most restrictive lines, the transient sample immediately proceeding it is accepted as a significant sample, and the procedure above is repeated from this point on.

ε is usually chosen between $0.01 - 0.04mV$. Larger ε causes the compression ratio to increase.

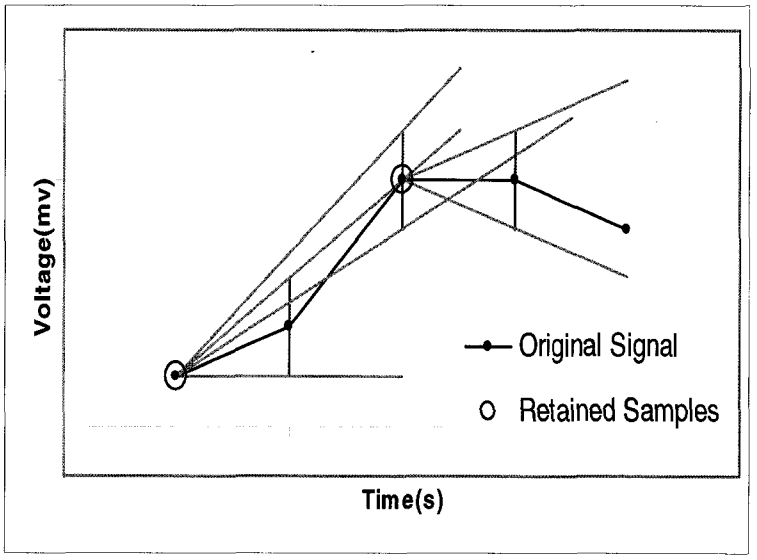


Figure 2.8: The FAN algorithm

2.3.2 AZTEC (Amplitude Zone Time Epoch Coding)

The AZTEC algorithm was originally developed for preprocessing real-time ECG's for rhythm analysis. It stands for Amplitude zone time epoch coding [3]. AZTEC is a preprocessing program which is activated by an interruption that converts input voltages to digital form.

The input to this algorithm is a set of original sample points which are then converted into plateaus and slopes. The threshold of this algorithm is a difference voltage value. Let v_i be the i^{th} sample following an initial sample of v_0. Let v_{max} and v_{min} be the maximum and minimum voltage values. Let $\{v_0, v_1, ..., v_m\}$ be a set of signal samples and K to be a voltage threshold. If $v_{max} - v_{min} \leq K$ for $\{v_0, v_1, ..., v_{n-1}\}$ and $v_{max} - v_{min} > K$ for $\{v_0, v_1, ..., v_n\}$ then the set $\{v_0, v_1, ..., v_{n-1}\}$ is called an AZTEC line. This line is stored by its value $\frac{1}{2}(v_{max} + v_{min})$ and its duration.

K is usually set to a number between $0.05 - 0.09mV$. Smaller K values will

decrease the compression ratio.

The AZTEC plateaus (horizontal lines) are produced by zero-order interpolation. When a signal of higher frequency and amplitude such as the QRS begins[1], the voltage samples will change rapidly, and lines of short duration will be formed. A series of lines, each containing four samples or less, is considered to be adequately represented by a constant rate of voltage change, or slope, as long as the voltage difference between adjacent lines does not change sign. The slope is terminated by a line longer than four samples or a change in signs (Figure 2.9).

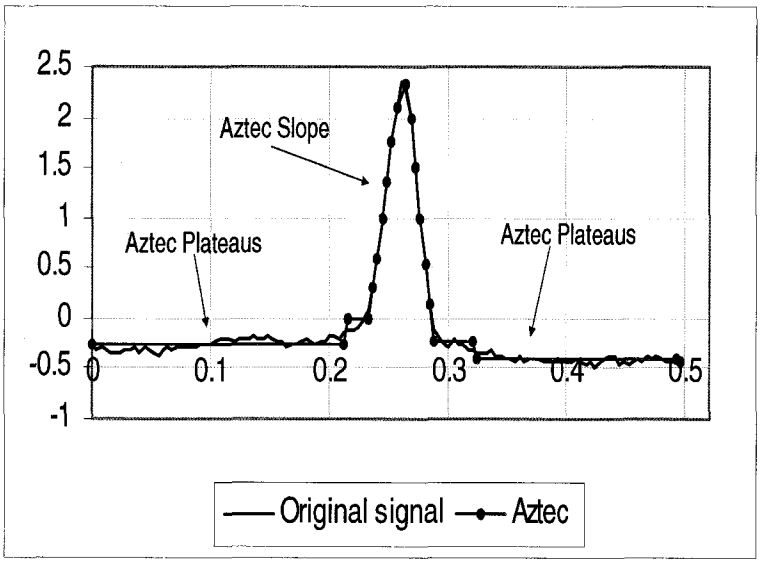


Figure 2.9: The Aztec algorithm

Even though the AZTEC provides a high data reduction ratio, the reconstructed signal has poor fidelity mainly because of the discontinuity (step-like quantization) of the waves.

[1]frequency of 150-250Hz

2.3.3 TP (Turning point)

The Turning Point algorithm processes three data points at a time: a reference point (y_0) and two consecutive data points (y_1 and y_2) [22].

The algorithm attempts to retain either y_1 or y_2 depending on which point preserves the slope of the original three points. The circled points in Figure 2.10 are selected by the TP algorithm.

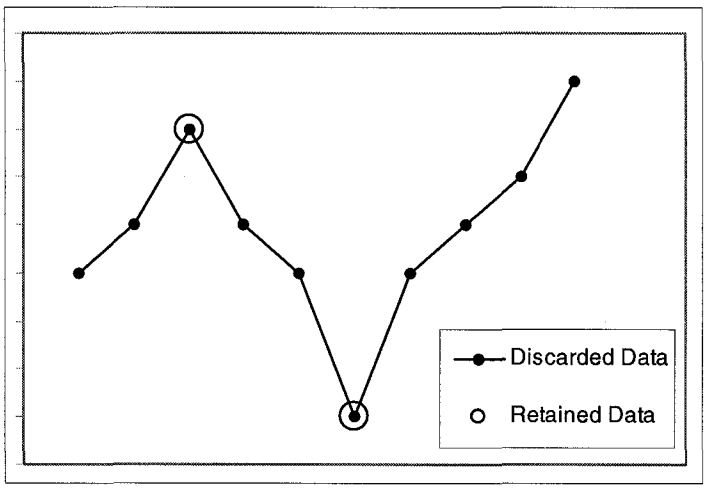


Figure 2.10: The TP algorithm

There are nine possible arrangements for three consecutive points as it is shown in Figure 2.11. If $(y_1 - y_0)(y_2 - y_1) \geq 0$ then y_2 is selected, otherwise y_1 is chosen. Table 2.1 shows the two patterns which y_1 is selected. In this method, only the amplitudes are to be stored but not their locations, therefore there are some local errors.

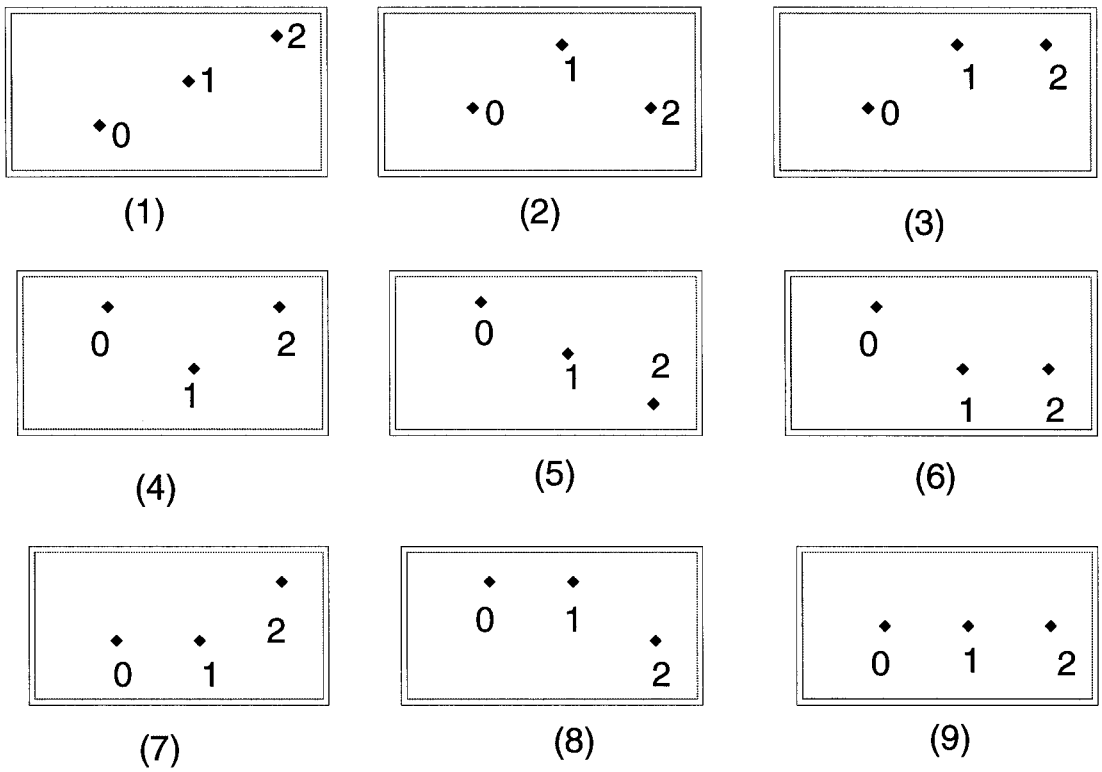


Figure 2.11: Nine arrangements of three points.

2.3.4 CORTES (The Coordinate Reduction Time Encoding System)

CORTES algorithm is a combination of the AZTEC and TP algorithms. TP preserves important attributes of the *QRS* complexes. However, it provides the same amount of data reduction to lower information signal areas, such as the P and T waves. On the other hand, AZTEC has great data reduction properties, adapts its coding mode to the nature of the signal region itself, and eliminates baseline small waves.

CORTES applies the TP algorithm to the high frequency regions (*QRS* complexes) and the AZTEC algorithm to the lower frequency regions (Figure 2.12) and the isopotential regions of the ECG signal where great data reduction takes place with little

Pattern	Sign of $y_1 - y_0$	Sign of $y_2 - y_1$	Sign of $(y_1 - y_0)(y_2 - y_1)$	Point choice
1	+	+	+	y_2
2	+	-	-	y_1
3	+	0	0	y_2
4	-	+	-	y_1
5	-	-	+	y_2
6	-	0	0	y_2
7	0	+	0	y_2
8	0	-	0	y_2
9	0	0	0	y_2

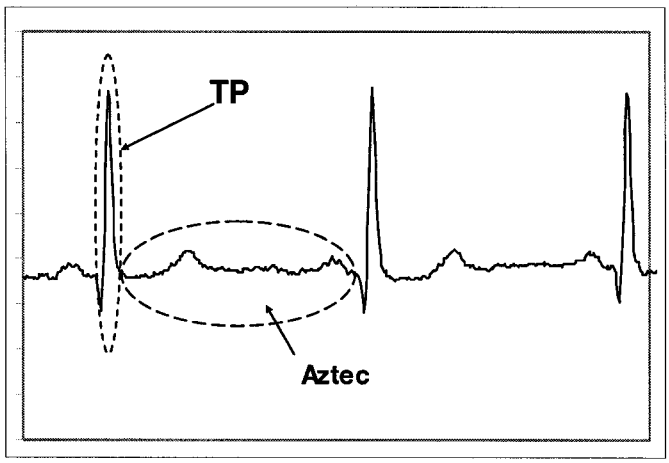


Figure 2.12: The CORTES algorithm

distortion [2]. CORTES is a fast algorithm particularly suited for microprocessor-based, real-time ECG analysis.

2.3.5 TRIM (Turning point/recursive improvement)

This algorithm is an improved version of Turning Point algorithm by reducing the number of selected points [20, 21]. The threshold of this algorithm is a voltage value. It uses the Turning Point algorithm to find the most significant points. The significance of each point is calculated as:

$$S = k_1|\Delta T_{min}| + k_2|\Delta V_{min}| - 1$$

ΔT_{min} is the time interval to the nearer of two adjacent turning points and ΔV_{min} is the voltage difference of the two turning points.

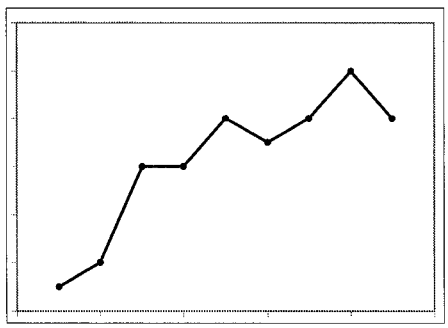
Suitable values for k_1 and k_2 are chosen[2]. The significant points which have a positive value of S and $\Delta V_{min} \geq 80\mu v$ are selected. The next phase of TRIM applies the RPC(Recursive Partitioning Compression) approach. TP uses the RPC to find the points between two turning points. At this stage a line segment is represented by linear interpolation of its two end points. Then the most distant point os the signal from this line is selected and if this distance is larger than the threshold, the line segment is replaced by two segments. Figure 2.13 shows how the algorithm works. The tolerance σV is a function of the length of the segment to be partitioned, ΔT, and the median output rate:

$$\sigma V = \frac{k_3(\Delta\overline{T}+\Delta T)}{\Delta T.\Delta\overline{T}}$$

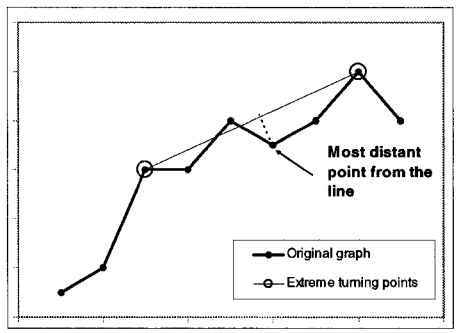
2.3.5.1 Douglas-Peucker Algorithm

This is a famous algorithm similar to TRIM [6]. For a set of sample points $Y = \{y_1, y_2, ..., y_N\}$, the algorithm looks for a set $Y' \subseteq Y$ within a certain distance threshold. It considers a line joining the first and last points in the sequence. Then the remaining points are tested for closeness to that line. If there are points further than a specified threshold, $\epsilon > 0$, away from the line, then the point furthest from it is added to Y'. Using recursion, this process continues for each line of the current guess

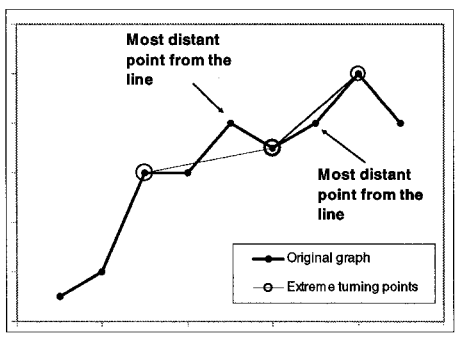
[2] $k_1 = 10sec^{-1}$ and $k_2 = 10^6 v^{-1}$



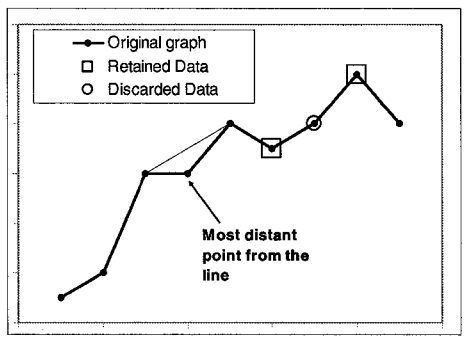
(a) A section of signal



(b) Detecting extreme turning points after preprocessing



(c) Partitioning the interpolated line



(d) Selecting the points

Figure 2.13: The TRIM algorithm

until all points of the original sequence are within the threshold of the simplification (Figure 2.14).

2.3.6 AZTDIS

This algorithm has two phases [30]. The first phase has a coarse data reduction, while the second phase reduces data significantly. The first one is similar to the AZTEC algorithm, but it only generates the plateaus.

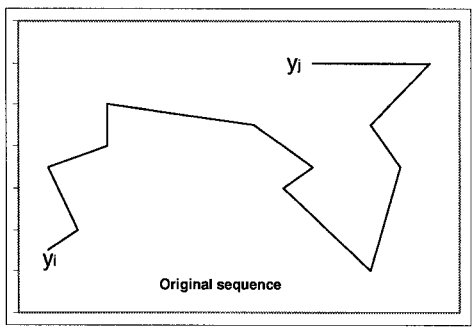
It saves the duration and the slope of each line. If the next sample changes direction, the algorithm uses a linear interpolation to recursively reconstruct the middle points (Figure 2.15) and uses the threshold δ to decide which points to keep.

A line is drawn between two selected points by AZTEC. The value of y_m can be calculated easily. Then the displacement between the encoded sample (x_m, y_m) and original sample (x_m, y_{i+1}) is $\delta = |y_m - y_{i+1}|$. If $\delta > \epsilon$, then the point (x_m, y_{i+1}) is selected as a significant sample and the line connecting y_i and y_{i+2} is broken into two segments connecting y_i to y_{i+1} and the segment y_{i+1} to y_{i+2}. This process will recursively continues.

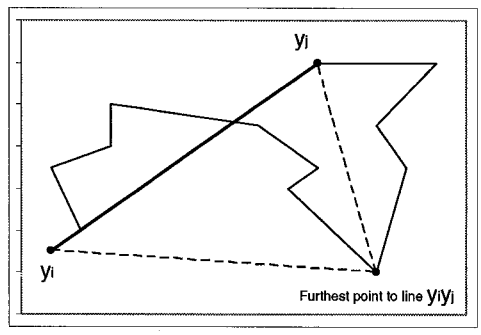
2.3.7 CCSP (Cardinality Constrained Shortest Path algorithm)

The input of this algorithm is not a voltage value or an error threshold. Instead it takes a compression ratio as its input and produces output which meets the required compression ratio minimizing the error [23]. By defining the desired compression ratio which leads to the total number of points in the compressed diagram, this algorithm produce a graph including those points with the minimum possible error.

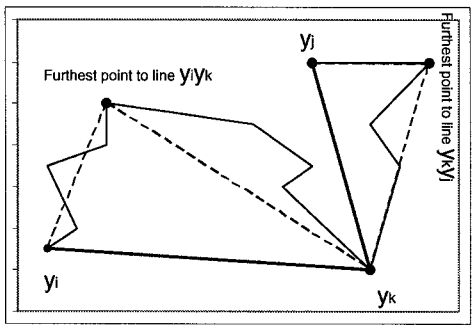
The goal is to find the path with minimum error between the first and last point. The path with minimum error can be found by RMS (Root Mean Square) between two points.



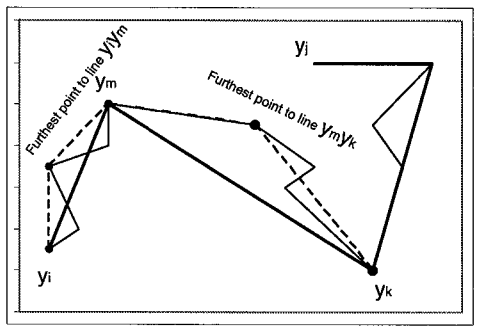
(a) original sequence



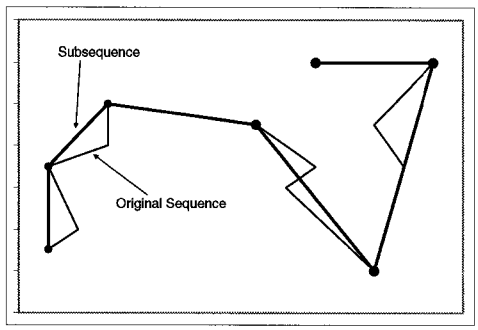
(b) The furthest point from the line connecting two end points



(c) Retaining more points



(d) Retaining more points



(e) Original sequence and retained subsequence

Figure 2.14: Douglas-Peucker Algorithm

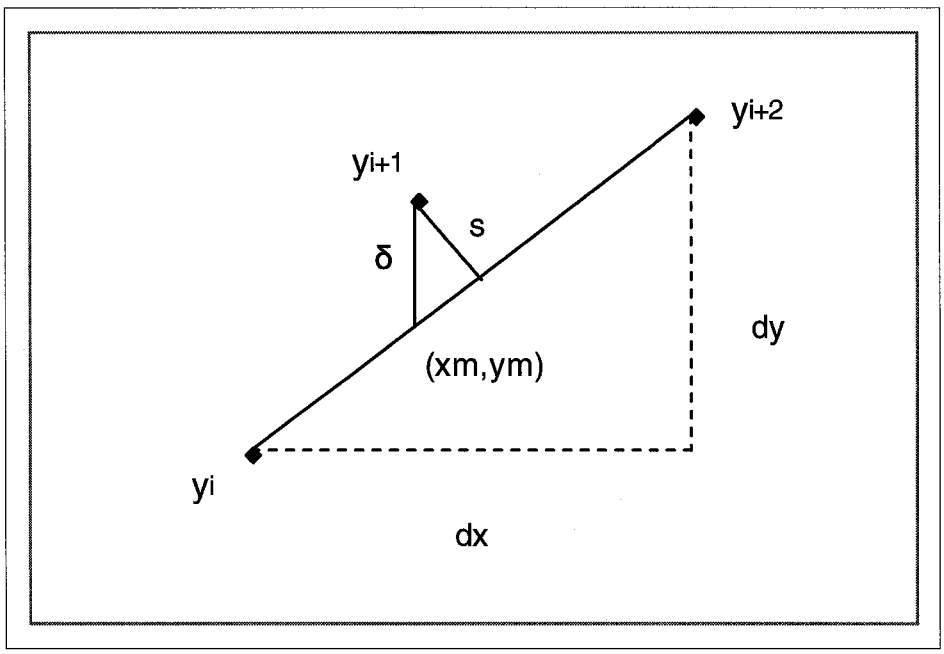


Figure 2.15: The AZTDIS algorithm-Second phase

A line is drawn between in P_i and P_j. The square root of the summation of the *distances* between each point and the line $\overline{P_iP_j}$ is called RMS. *Distance* here means the length of the line that is drawn parallel to the y axis until it intersects to the $\overline{P_iP_j}$ and it is not the real distance. This RMS [3] value is considered the error in this algorithm. The path will be picked based on the number of points to be chosen in a path and the minimum error between $\overline{P_iP_j}$.

CCSP constitutes a dynamic programming solution to the shortest path problem it defines. This algorithm will be intensively discussed in Chapter 4.

[3]In the original algorithm, the path with the least RMS is called *shortest path.*

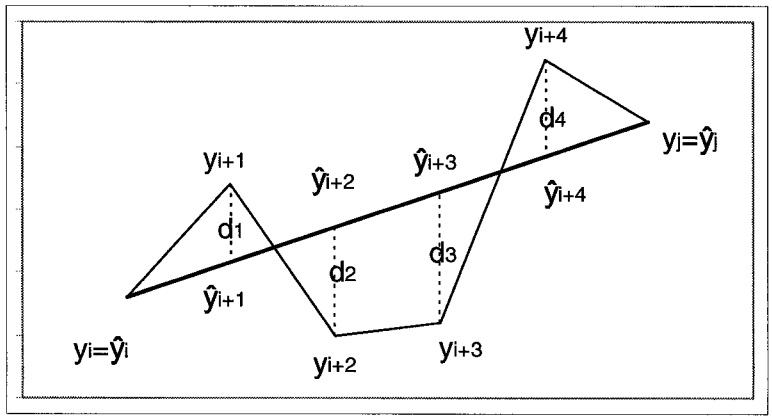


Figure 2.16: RMS in CCSP Algorithm

2.3.8 Summary of lossy algorithms

Table 2.2 shows a summary of the ECG data compression schemes from the original papers [11, 30, 9, 31, 17]. These schemes are tested in chapter 5 of this thesis.

The frequency in this table is the frequency which is usually used to test the scheme. The compression ratio (CR) is the number of selected points in the compressed set with respect to the total number of points in original set which is a number between 0 and 1.

Percent Root-mean-square Difference(PRD) indicates the error between two signals and can be calculated as follows [23]:

$$PRD = \sqrt{\frac{\sum_{n=1}^{N}(y_n - \hat{y}_n)^2}{\sum_{n=1}^{N}(y_n - \bar{y}_n)^2}} \times 100\% \tag{2.1}$$

Where y_n represents the original point, $\hat{y}_n$ indicates the reconstructed value of y_n and $\bar{y}_n$ is the mean value in the original set. PRD represents the error between two signals; the smaller the PRD values, the more similar signals together are achieved.

Even though some of these algorithms can be used in wireless applications, the quality of their compressed graph is not very good. From those which have the on line

Algorithm	Compression Ratio	PRD	Frequency	On-line capability
Fan	3:1	4%	250	yes
AZTEC	10:1	28%	500	yes
TP	2:1[a]	5.3%	200	yes
TRIM[b]	10:1	10%	120	No
CORTES	4.8:1	7%	200	yes
AZTDIS[c]	7.9:1	9.7%	360	No
CCSP	15:1	7.5%	500	No

[a]TP may reach a compression ratio less than 2:1 by repetition
[b]Requires preprocessing
[c]Requires preprocessing

Table 2.2: Summary of lossy ECG data compression techniques

capability, we have implemented Fan, Aztec, TP, and CORTES. However CCSP can not be used in on-line applications, it has been implemented because of the quality of the reconstructed signal. Signal preprocessing requires all the data to be available before running the algorithm which is impossible in on-line applications. Since it is required in AZTDIS and TRIM, they have not been implemented and studied in this thesis. The quality of the rest of the methods are evaluated in Chapter 5 in detail.

Chapter 3

Least Area method

Compression of ECG signals has traditionally been tackled by heuristic approaches, such as those described in Chapter 2. Time domain compression algorithms extract a subset of significant signal samples. Most algorithms suffer from the lack of ability to extract signal samples in a manner that guarantees the smallest reconstruction error.

On-line algorithms must process the data as data is received and their running time must allow them to process the data fast.

In the algorithm presented in this chapter, only a constant amount of data can be stored during the process. When the algorithm selects a point, then it can discard the previous points it has seen so far. A point is selected either when it is a turning point in a signal or the signal has reached its maximum possible threshold at that point. The algorithm provides high quality compression, it is very fast, does not need a large amount of memory to store the data and can be used in on-line applications.

3.1 Preliminaries

The term *Area* is used to defined an enclosed region between two curves $f(x)$ and $g(x)$ (Figure 3.1). $Area_{i,j}$ represents the area between two curves $f(x)$ and $g(x)$, and

lines $x = x_i$ and $x = x_j$.

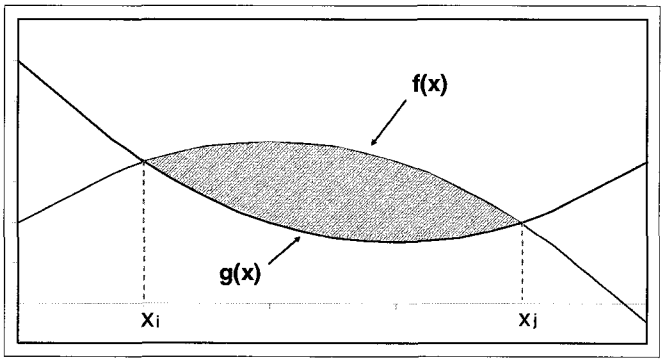


Figure 3.1: The area between two curves

The area between two signals can always be divided into triangles and trapezoids which are easy to calculate.

The intersection of the line drawn from each point y_k parallel to y axis and the line between start and end points ($\overline{y_{i+1}y_{i+5}}$ in Figure 3.2) gives $\hat{y}_k$. If the two curves do not intersect with each other rather than start and end points like Figure 3.2, the area is computed easily. For example the area of triangle $y_{i+1}y_{i+2}\hat{y}_{i+2}$ can be computed as:

$area_1 = \frac{b_1(x_{i+2}-x_{i+1})}{2}$

where: $b_1 = \hat{y}_{i+2} - y_{i+1}$

The area of trapezoids $y_{i+2}y_{i+3}\hat{y}_{i+3}\hat{y}_{i+2}$ can be calculated as:

$area_2 = \frac{(b_1+b_2)(x_{i+3}-x_{i+2})}{2}$

where: $b_1 = \hat{y}_{i+2} - y_{i+1}$ and $b_2 = \hat{y}_{i+3} - y_{i+3}$.

All the triangles and trapezoids can be calculated in the same way and their total summation gives the total area.

In Figure 3.3 the area of triangles $y_iy_{i+1}\hat{y}_{i+1}$ and $y_jy_{i+2}\hat{y}_{i+2}$ is calculated in the

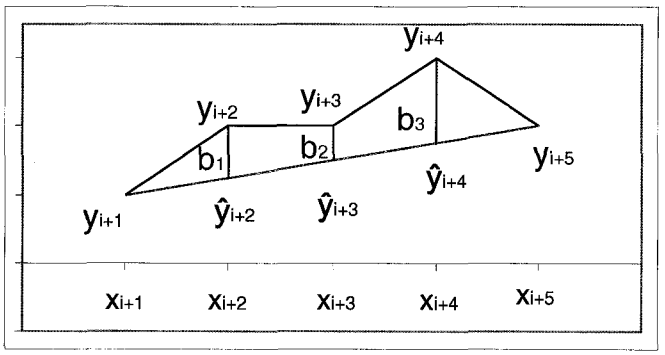


Figure 3.2: Calculating the area between two signals.

way described above.

However the area of triangles $y_{i+1}\hat{y}_{i+1}y_{12}$ and $y_{i+2}\hat{y}_{i+2}y_{12}$ can be calculated as $\frac{h_1 b_1}{2}$ and $\frac{h_2 b_2}{2}$, we can compute them without finding the y_{12} point as follows:

$$\frac{h_1}{b_1} = \frac{h_2}{b_2} \Rightarrow h_1 = \frac{b_1}{b_2} h_2$$

$$d_x = h_1 + h_2$$

$$d_x = \frac{b_1}{b_2} h_2 + h_2$$

Therefore the values of h_1 and h_2 are:

$$h_2 = \frac{b_2}{b_1 + b_2} d_x$$

$$h_1 = \frac{b_1}{b_1 + b_2} d_x$$

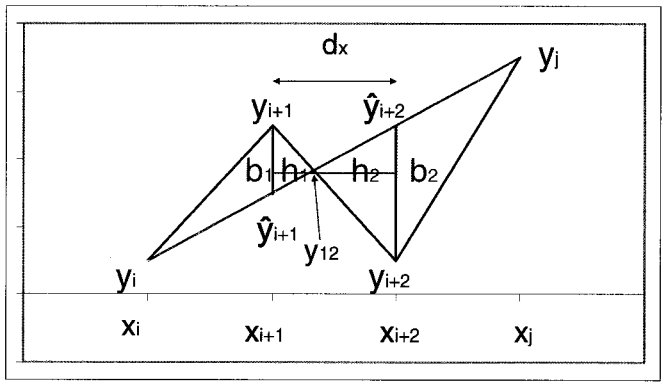


Figure 3.3: Calculating the area between two curves with more than two intersection points

Therefore the area of both triangles will be:

$$area = \frac{b_1^2 + b_2^2}{|b_1| + |b_2|} \frac{d_x}{2}$$

where:

$b_1 = y_{i+1} - \hat{y}_{i+1}$

$b_2 = y_{i+2} - \hat{y}_{i+2}$

and

$d_x = x_{i+2} - x_{i+1}$

3.2 Problem definition

Denote an ECG digital signal taken at constant time intervals by the sequence $Y = \{y_1, y_2, ..., y_N\}$. Let y_1 be the first point and N denote number of samples received during the execution time [23]. The input of the algorithm is an objective function

and a threshold ε_A which is described in Section 3.4.1. Let $Y' = \{y'_1, y'_2, ..., y'_M\}$ be a subsequence of Y and there exists a set of indices $C = \{c_1, c_2, ..., c_M\}$: $1 \leq c_i \leq N$ and $c_i < c_{i+1}$ and $c_1 = 1, c_M = N$ such that $y'_i = y_{c_i}$. We are looking for Y' such that the area between Y and Y' is less than the given ϵ_A. The reconstructed set is a set $\hat{Y} = \{\hat{y_1}, \hat{y_2}, ..., \hat{y_N}\}$ which can be calculated as follows:

$$\hat{y_i} = \begin{cases} y'_k & \text{if } i = c_k, \\ f(y'_k, y'_{k+1}) & \text{such that } c_k < i < c_{k+1}. \end{cases} \tag{3.1}$$

$f(y'_k, y'_{k+1})$ is linear interpolation between y'_k and y'_{k+1}. Therefore each point on that line at equal time intervals between these two points represent the values of $\hat{y_n}$ in the reconstructed graph.

The interpolation provides a piecewise approximation to the original signal. Between two sample amplitudes corresponding to two succeeding elements of subset C, different linear functions are used to reconstruct the original signal. The choice of C will thus have vital importance for the quality of our signal approximation.

Figure 3.4 shows a part of the original and reconstructed signals. Note that there are a couple of $\hat{y_i}$s which are equal to y'_is.

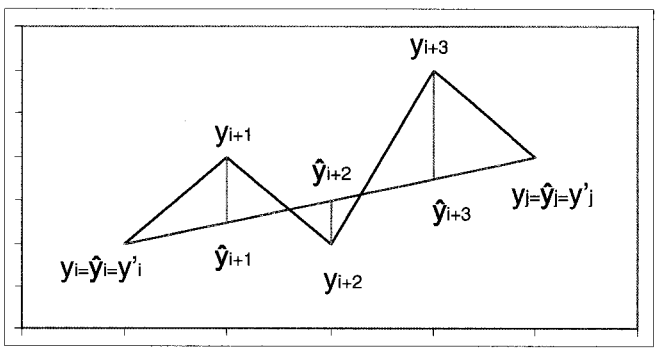


Figure 3.4: Illustration of y_i, y'_k and $\hat{y_i}$ (Black line: Original signal- Red line: Reconstructed signal)

3.3 Definitions

Assume the graph G consists of N points, $G = \{y_1, y_2, ..., y_N\}$. Denote by Δt, the time interval between samples y_i and y_{i+1} where $1 \leq i < N$. The graph G is continuous between t_1 and $t_N = t_1 + N\Delta t$

P_i represents the i^{th} point of a set with x_i and y_i values. Because all the x intervals are equal in this thesis, x_i can be omitted and y_i can represent the point.

Definition 1. *The point from which a line is drawn is called the starting point.*

Definition 2. *Two points are neighbors if there are no other points between them.*

Corollary 1. *Each point has at most two neighbors.*

Definition 3. *For each three consecutive points, y_i, y_{i+1} and y_{i+2}, if $(y_i - y_{i+1})(y_{i+1} - y_{i+2}) \leq 0$, y_{i+1} is called a turning point.*

3.3.1 Error Definition

The purpose of our algorithm is to keep the area between the original and reconstructed signals less than an *Area threshold.*

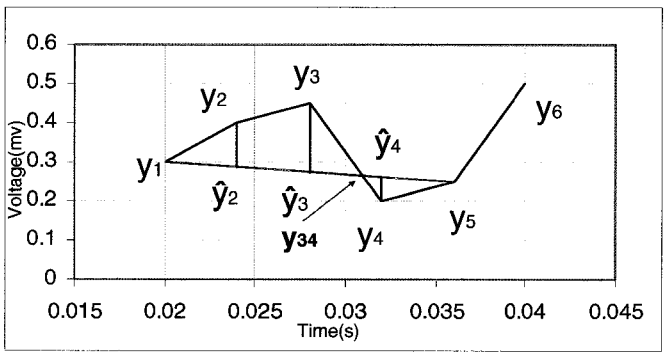
$$\epsilon_A \geq Area_{1,N} \tag{3.2}$$

Definition 4. *The total possible Area Threshold between the original and reconstructed graphs at the same time intervals is ϵ_A such that equation 3.2 holds.*

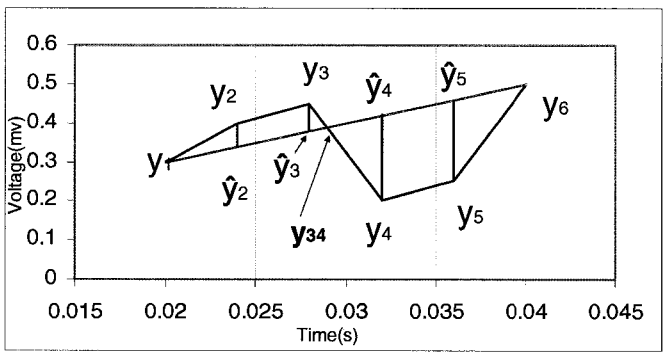
Assumption 1. *Each area is considered to be a positive value so as to keep the Area Threshold growing overtime.*

Suppose the first point in a graph is y_i, the last point is y_k and the second last point is y_{k-1} and $Area_{i,k-1} \leq \epsilon_A \leq Area_{i,k}$. Therefore y_{k-1} is selected.

This is illustrated in Figure 3.5 where y_1 is the first point selected in the graph G and and y_5 is the last point which satisfies the condition has been selected.



(a) The area between the line $\overline{y_1 y_5}$ and the original graph



(b) Adding next point

Figure 3.5: Generating different areas

Total area $= a_1 + a_2 + a_3 + a_4 + a_5$

Where y_{34} is the intersection point between two lines $\overline{y_3 y_4}$ and line $\overline{y_1 y_5}$. The other terms are defined in Table 3.1.

Different Areas			
Figure	Area	Type	Value
3.5(a)	a_1	triangle $y_1 y_2 \hat{y}_2$	0.00022
	a_2	trapezoid $y_2 y_3 \hat{y}_3 \hat{y}_2$	0.00052
	$a_3 + a_(4)$	triangle $y_3 y_{34} \hat{y}_3 + y_{34} y_4 \hat{y}_4$	0.00025
	a_5	triangle $y_4 y_5 \hat{y}_4$	0.000025
	Total Area		0.001015
3.5(b)	a_1	triangle $y_1 y_2 \hat{y}_2$	0.0001
	a_2	trapezoid $y_2 y_3 \hat{y}_3 \hat{y}_2$	0.00012
	$a_3 + a(4)$	triangle $y_3 \hat{y}_3 y_{34} + y_{34} \hat{y}_4 y_4$	0.000355
	a_5	trapezoid $y_4 y_5 \hat{y}_5 \hat{y}_4$	0.00086
	a_6	triangle $\hat{y}_5 y_5 y_6$	0.00042
	Total Area		0.001855

Table 3.1: Triangle and non-triangle areas for Figure 3.5(a) and 3.5(b)

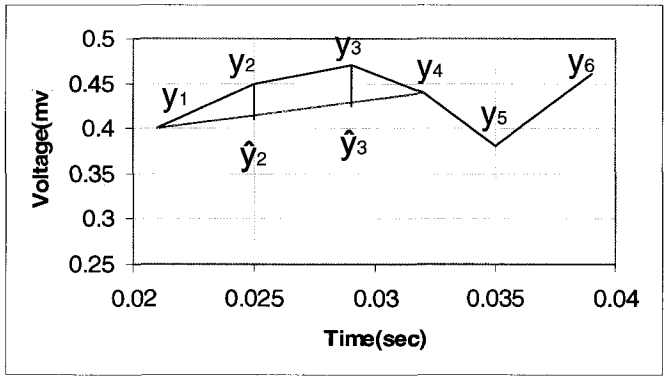
In Figure 3.5(a) suppose the area threshold $\epsilon_A = 0.0015$ is used and

Total area$= 0.001015 \leq \epsilon_A$.

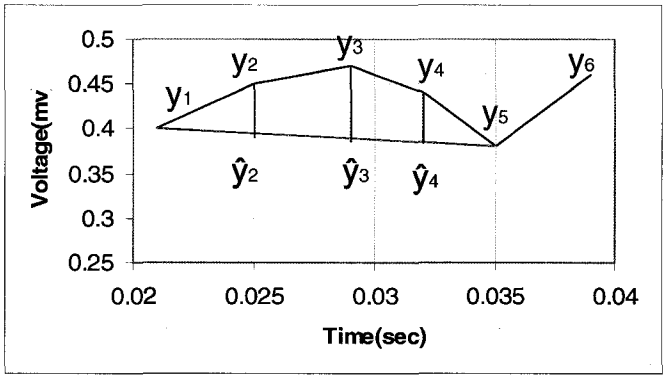
Since in Figure 3.5(b) the end point is y_6, all the areas are different than Figure 3.5(a) and *Total area* $= 0.001855 \geq \epsilon_a$. Since y_5 is the last point which fulfills equation 3.2, it is selected as the next starting point.

In Figure 3.5 there are more than two intersection points between two signals. Figure 3.6 shows another example which there is only two intersection points between two signals. Suppose $\epsilon_A = 0.001$. In Figure 3.6(a), there are three areas and in Figure 3.6(b) four areas are generated which their values are given in Table 3.2:

In 3.6(b), adding y_5 has violated equation 3.2 and the *Total area* has become larger than the threshold. Therefore, y_4 is chosen as the next starting point.



(a) The area between the line $\overline{y_1 y_5}$ and the original graph



(b) Adding next point

Figure 3.6: Non-triangle areas between the original and compressed graphs.

3.4 Selecting the sequence of Y'

In this section we present the decision mechanism which is used to select sample points from the original signal in order to create the subsequence Y'. Two different thresholds are required in order to select the subsequence adequately. One is *Area Threshold*, ϵ_A, and the other is *Voltage Threshold*ϵ_V which is used for Turning points only.

Different Areas			
Figure	Area	Type	Value
3.6(a)	a_1	triangle $y_1y_2y_{12}$	0.00024
	a_2	trapezoid $y_2y_3y_{12}y_{13}$	0.00048
	a_3	triangle $y_3y_4y_{13}$	0.00018
	Total Area		0.0009
3.6(b)	a_1	triangle $y_1y_2y_{12}$	0.00024
	a_2	trapezoid $y_2y_3y_{12}y_{13}$	0.00048
	a_3	triangle $y_3y_4y_{13}$	0.00018
	a_4	triangle $y_4y_5y_6$	0.000665
	Total Area		0.001565

Table 3.2: Triangle and non-triangle areas for Figure 3.6

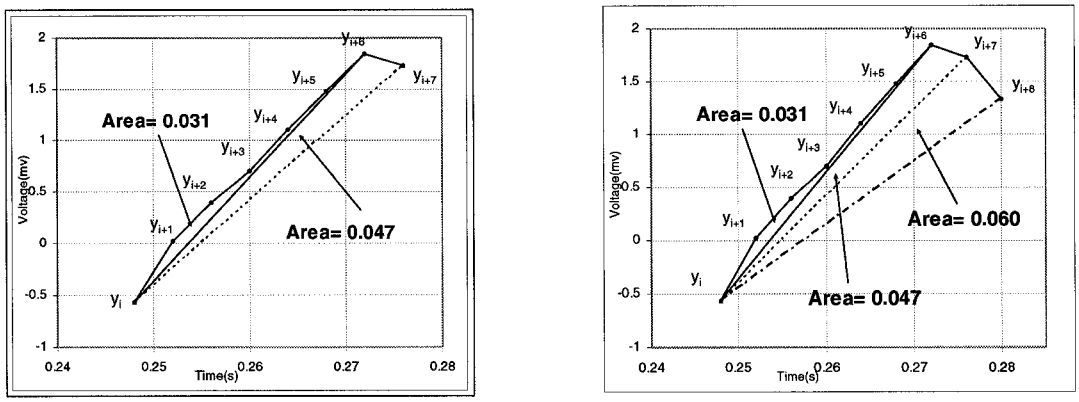
3.4.1 Area threshold

The first point in the original signal is always selected as the starting point. A line is drawn between a starting point, y_i, and the next point to be processed, y_j, which will be the second point after starting point. If $Area_{i,j} \leq \epsilon_A$, then the process continues with y_{j+1}. If $Area_{i,j} > \epsilon_A$, y_{j-1} is added to the compressed set Y' and all the information about the points between y_i and y_{j-1} is deleted. y_{j-1} becomes the next starting point and the procedure continues.

3.4.2 Turning points

Critical turning points in an ECG are P, Q, R, S, T and U. The *Area threshold* can not select all the critical turning points alone. For example if the line has a sharp slope, like the line between points **Q** and **R** in the **QRS** complex, the area between the line and the original graph would be very small and would likely not violate the *Area threshold* condition.

Figure 3.7 shows a section of the **QRS** complex in which point y_{i+6} represent the point **R** of the signal. y_{i+5} is chosen if by adding y_{i+6}, the total created area between the line $\overline{y_{i+1}y_{i+6}}$ and graph G is greater than ϵ_A.



(a) $Area_{i,i+7}$ less than ϵ_A

(b) $Area_{i,i+8}$ more than ϵ_A

Figure 3.7: Possible loss of point y_6

In Figure 3.7(a), ϵ_A is 0.05 and y_{i+6} is a critical point like point **R** which is not selected in this case because $Area_{i,i+7} < \epsilon_A$. In Figure 3.7(b), $Area_{i,i+7} = 0.047$ while $Area_{i,i+8} = 0.060$ and $Area_{i,i+8} > \epsilon_A$. Therefore point y_{i+7} is chosen and point y_{i+6} is lost.

Since some of the important turning points like **R** in and ECG graph may not be captured by the *Area Threshold*, the slope of these points with their neighbors must

be calculated. If at a point the sign of slope of the line changes, it is a turning point. But in this way, every single turning point will be chosen. Therefore we need a second threshold to determine if this is a valuable turning point or not.

3.4.2.1 Voltage threshold

Figure 3.8(a) shows a high frequency section of an ECG graph. The absolute value of the voltage difference between two neighbors and changes in slope are presented in Figure 3.8(b) and 3.8(c).

P_j represents point **R** of a **QRS** complex in Figure 3.8(a) which is a turning point and P_i is another point in the same complex which is not a turning point. The differences between the value of y_j and its neighbors are 0.395 and 0.393. Figure 3.8(c) shows that the slope of two segments $\overline{y_{j-1}y_j}$ and $\overline{y_jy_{j+1}}$ changes at P_j. P_i has more voltage difference with its neighbors P_{i-1} and P_{i+1} than what P_j has with its neighbors P_{j-1} and P_{j+1}, but the slope sign of the two segments $\overline{y_{i-1}y_i}$ and $\overline{y_iy_{i+1}}$ does not change at P_i.

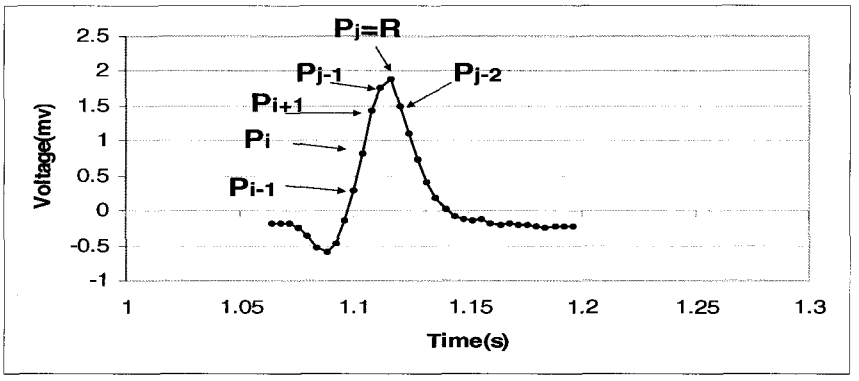
Figure 3.9(b) and 3.9(c) show the voltage difference between two neighbors and changes in slope for the lower frequency part of the signal shown in Figure 3.9(a).

Point P_i denotes a turning point in Figure 3.9(a). The sign of the slope changes at P_i in Figure 3.9(c) while the difference between y_i and its neighbors y_{i-1} and y_{i+1} is 0.01 and 0.005.

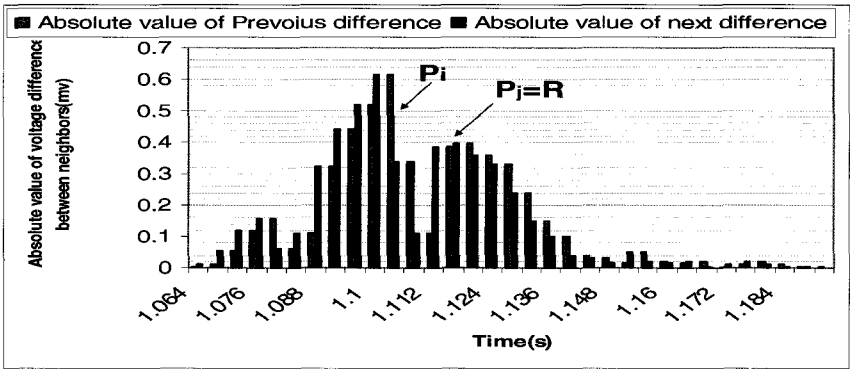
These Figures show that the voltage difference between two neighboring points in the low frequency part of a graph can be very smaller than the voltage difference between two neighbor points in high frequency areas.

We have used the experiment shown in the Figures 3.8 and 3.9 to introduce the *Voltage threshold.* This threshold, ϵ_V, allows the turning points such as **R** to be captured while reducing the small steady changes mostly in lower frequency parts.

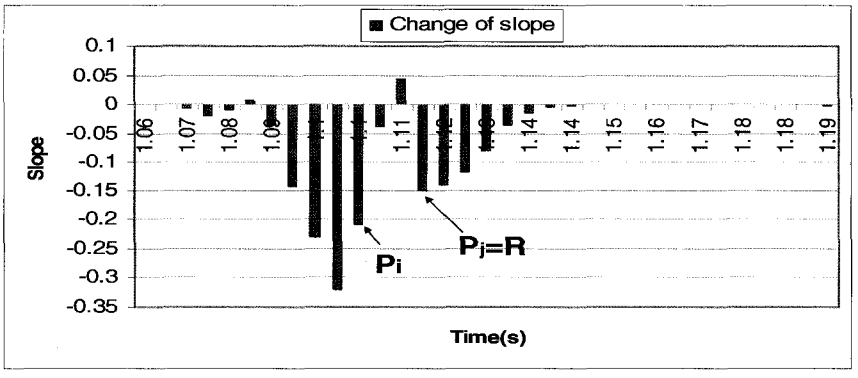
1-Experimental ϵ_V value



(a) PAF Prediction Challenge Database

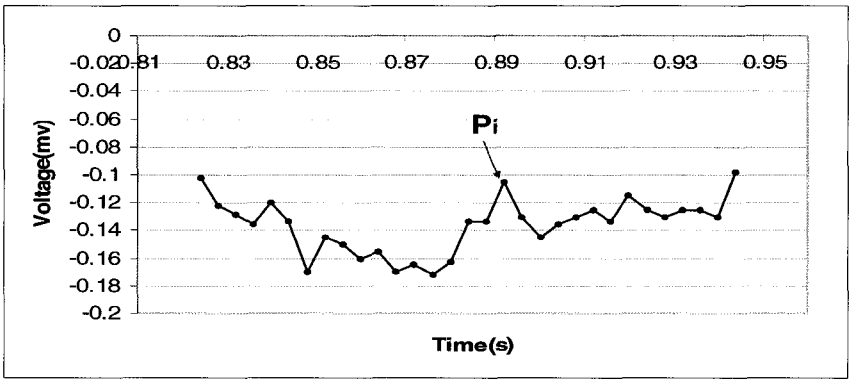


(b) Voltage difference

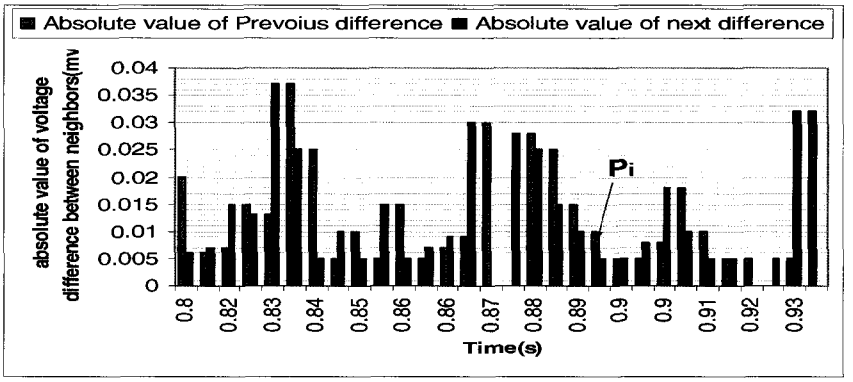


(c) Slope changes

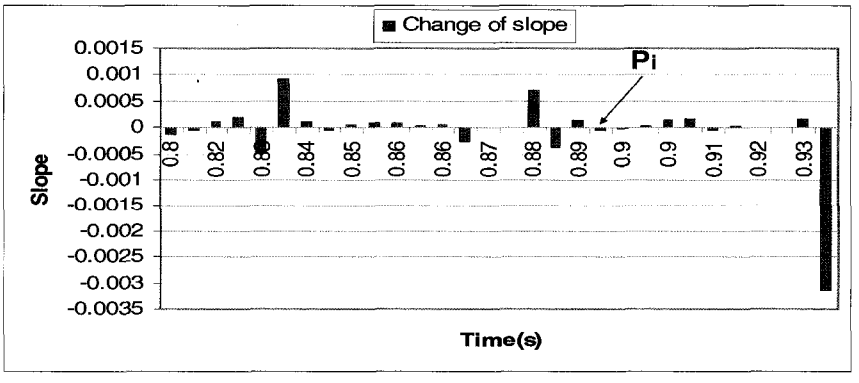
Figure 3.8: High frequency part of ECG



(a) MIT-BIH Normal Sinus Rhythm Database



(b) Voltage difference



(c) Slope changes

Figure 3.9: Low frequency part of ECG

Table 3.3 shows the average absolute value of voltage difference between two neighbors in high and low frequency parts from four different signal databases. The high and low frequency parts have been chosen randomly. These average values are calculated based on a specific part of the signal considered high voltage and low voltage.

Average voltage difference		
Database	High Frequency	Low Frequency
PAF	0.17875	0.00875
	0.185	0.009545455
	0.166666667	0.0085
	0.17432	0.00889
Average	0.176184167	0.008921364
AF Termination	0.387333333	0.019
	0.493	0.0366
	0.5394	0.0424
	0.5096	0.019563107
Average	0.482333333	0.029390777
AHA database	0.027157895	0.034927778
	0.178285714	0.03739779
	0.0245625	0.048994624
	0.1878	0.016264706
Average	0.104451527	0.034396224
BIDMC	0.76	0.047093023
	0.598333333	0.035526316
	0.633333333	0.014625
	0.356666667	0.014069767
Average	0.587083333	0.027828527

Table 3.3: Experimental value of ϵ_V

The pattern of these ϵ_Vs are quite similar. For the **QRS** complex the changes are quite sudden and therefore the average ϵ_V value high. The lower frequency part does not have a sharp and high change in voltage values.

The average voltage difference for high frequency area is 0.33 and for low frequency areas is 0.02. The value of ϵ_V can be chosen a number between these two averages as

far as it is not very close to the boundaries. In this thesis, the value 0.08 is assigned to ϵ_V in order to start the algorithms for frequency between 250 and 1000. Because this is used in on-line applications, this threshold is updated during execution to capture any sudden changes which might happen while recording an ECG.

Before the actual usage of ϵ_V, it is updated by averaging a few high frequency sections of the signal. However, if there is a sudden change in the signal, it would replace the previously saved ϵ_V value.

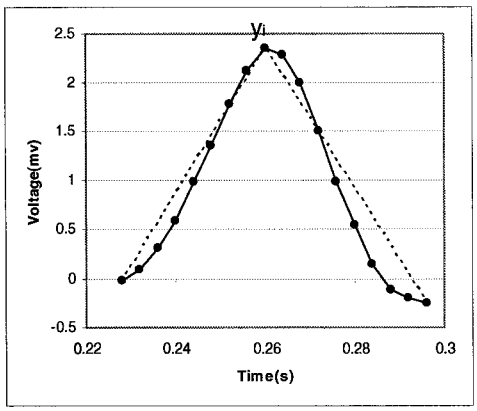
Before this threshold can be used in algorithm, its impact on different sections of the signal must be studied in order to prevent it from forcing the algorithm to select unnecessary points. ϵ_V must only be used to find the high frequency turning points which may not be selected by ϵ_A.

2- Effect of ϵ_V on High Frequency Sections

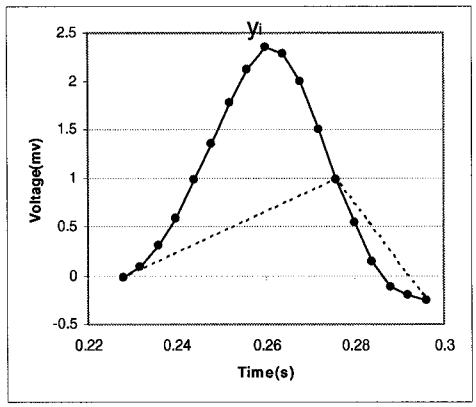
The ϵ_V value must be assigned from the voltage difference of high frequency sections. Figure 3.10(b) shows if the ϵ_V is much greater than the real voltage difference, then the critical points are lost. Suppose y_i is a critical turning point. If $\epsilon_V > |y_i - y_{i-1}|$ and $\epsilon_V > |y_i - y_{i+1}|$, then y_i is not selected and since it has not been fulfilled the ϵ_A condition as well, it is lost.

Suppose ϵ_V is smaller than the real voltage difference in high frequency section. We know that the voltage difference in the high frequency sections are greater than voltage differences in any other sections. The small value of ϵ_V forces the algorithm to select more points than it should, because it selects many non critical turning points. As a result, it does not allow the compression ratio to be less than a certain value regardless of the *Area threshold* value. An example of this case is shown in Figure 3.11(b). With $\epsilon_V = 0.08$ and $\epsilon_A = 0.35$, a compression ratio of 0.0811 can be achieved. If we do not change the ϵ_A, but reduce ϵ_V to 0.008 which is smaller than average ϵ_Vs in lower frequencies of Table 3.2, the compression ratio increase to 0.4189 which is more than 5 times what it should achieve.

3- Effect of ϵ_V on Low Frequency Sections



(a) ϵ_V=0.08; ϵ_A=0.8; Compression ratio=0.16



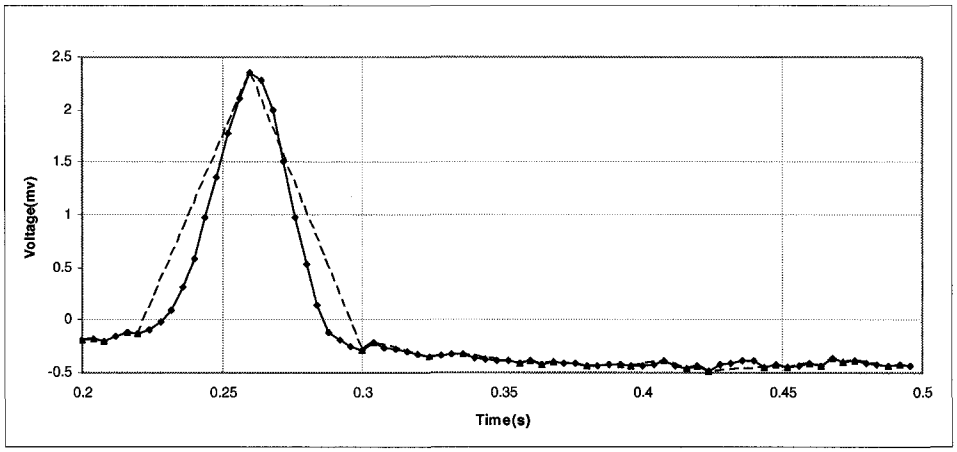
(b) ϵ_V=0.8; ϵ_A=0.8; Compression ratio=0.16

Figure 3.10: Effect of ϵ_V on High Frequency Sections

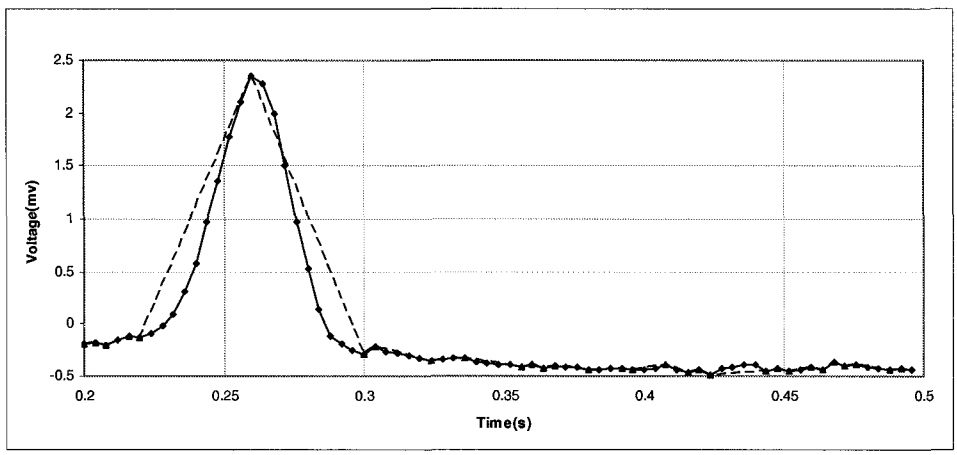
High values of ϵ_V usually do not affect low frequency sections of the signal, because the voltage difference in these parts are much smaller than ϵ_V.

Small values of ϵ_V may cause too many points to be selected in lower frequency parts. Table 3.4 shows the voltage difference between neighbors of Figures 3.12(a) and 3.12(b). Since $\epsilon_V = 0.08$ in Figures 3.12(a), *Voltage threshold* does not affect the result of *Area threshold* decision. By changing the value of ϵ_V to 0.008 in Figures 3.12(b), the voltage difference for all of the neighbors from P_{i-2} to P_{i+2} become greater than ϵ_V and therefore it forces all of these points to be selected.

Since ϵ_V has a greater impact on selecting points than ϵ_A, its value is updated during the first few seconds of receiving data. The first five R points are found and the absolute voltage difference between them and their neighbors are calculated and the average of this value is replacing the ϵ_V.

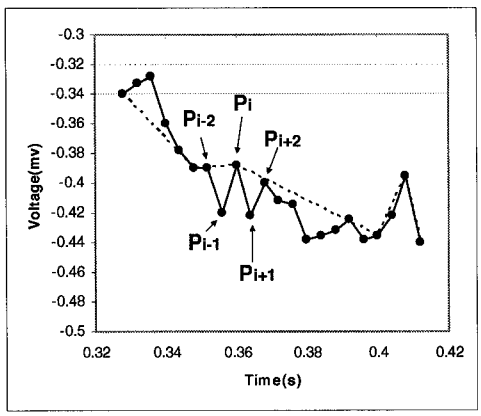


(a) ϵ_V =0.08 ; ϵ_A =0.35;Compression ratio =0.0811

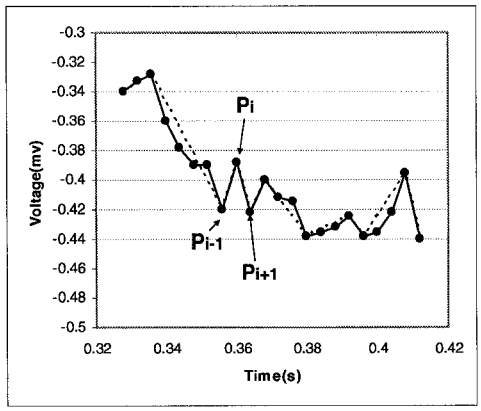


(b) ϵ_V =0.008 ; ϵ_A =0.35;Compression ratio =0.4189

Figure 3.11: Overcoming ϵ_V value on actual ϵ_A



(a) ϵ_V=0.08; ϵ_A=0.03; Compression ratio=0.22



(b) ϵ_V=0.008; ϵ_A=0.03; Compression ratio=0.22

Figure 3.12: Effect of ϵ_V on Low Frequency Sections

3.4.2.2 Critical turning points

In order to determine which turning points are critical two conditions must be checked. If both condition are met, then that point is flagged as a critical point which must be included in the compressed set.

The two conditions are:

1. Point P_i must be a turning point.
2. It must satisfy the ϵ_V threshold as follows:

$$\epsilon_V < |y_i - y_{i-1}|$$

OR

$$\epsilon_V < |y_i - y_{i+1}|$$

Segment	Voltage difference
$P_{i-2}P_{i-1}$	0.03
$P_{i-1}P_i$	0.035
P_iP_{i+1}	0.037
$P_{i+1}P_{i+2}$	0.022

Table 3.4: Absolute value of voltage difference between neighbors in Figures 3.12(a) and 3.12(b).

3.5 Least Area Algorithm

We designed an algorithm which uses the least area objective function. Let $Y = \{y_1, y_2, ..., y_N\}$ represent the original sequence of N points and the compressed set be $Y' = \{y'_1, y'_2, ..., y'_M\}$ if there exists a set $C = \{c_1, c_2, ..., c_M\}$: $1 \leq c_i \leq N$ and $c_i < c_{i+1}$ and $c_1 = 1, c_M = N$ such that $y'_i = y_{c_i}$ and the area between Y and Y' is less than ϵ_A. The reconstructed set is a set $\hat{Y} = \{\hat{y_1}, \hat{y_2}, ..., \hat{y_N}\}$.

Suppose there are a set of points between P_i to P_j. We search for a point P_k such that $i < k < j$ and $Area_{i,k} \leq \epsilon_A$ and either P_k is a turning point or $Area_{i,k+1} > \epsilon_A$.

The first step in solving the LA problem is to draw a line between the first (P_i) and the next proceeding point (P_j) of a signal and then to find the area that is generated between this line and the original graph. To calculate $Area_{i,j}$, both x and y values of the points between P_i and P_j are required as described in Algorithm 1.

$Area_{i,j}$ then is checked with the *Area threshold* condition. If it meets the condition then $Area_{i,j+1}$ is calculated. Every time a new point is added to the process, the areas are different and therefore the areas must be calculated again.

The process of adding a point stops either when the *Area threshold* is not met by the last point or it is a turning point. Suppose the first point is P_i, the last point is P_j and the second last point is P_{j-1}. If $Area_{i,j} > \epsilon_A$, point P_{j-1} is selected to add compressed set. If $Area_{i,j} < \epsilon_A$, P_j should be checked if it is a turning point or not. Only the y value of P_j is used to determine if it is a valuable turning point as it is

Algorithm 1 Calculate Area Algorithm

1: INPUT: P_i and P_j
2: OUTPUT: $Area_{i,j}$
3: $Area_{i,j} \leftarrow 0$
4: **for** $j \leftarrow i$ to $k-1$ **do**
5: $dy_1 = y_j - (y_j + \frac{y_k - y_i}{k-i}(j-i))$
6: $dy_2 = y_{j+1} - (y_j + \frac{y_k - y_i}{k-i}(j-i))$
7: **if** $dy_1 dy_2 \geq 0$ **then**
8: $Area_{i,j} \leftarrow Area_{i,j} + |\frac{dx(dy_1+dy_2)}{2}|$
9: **else**
10: $Area_{i,j} \leftarrow Area_{i,j} + \frac{y_1^2+y_2^2}{|y_1|+|y_2|}\frac{dx}{2}$
11: **end if**
12: **end for**
13: return $Area_{i,j}$

described in Algorithm 2.

In Algorithm 2, the sign of the equation in line 3 can determine if it is a turning point. The second threshold in line 4 decides whether or not the point is critical.

Algorithm 2 LA Critical Turning Point Algorithm

1: INPUT: y_{j-1}, y_j and y_{j+1}
2: OUTPUT: true if y_j is s turning point OR false if y_j is not s turning point
3: **if** $(y_{j-1} - y_j) * (y_j - y_{j+1}) \leq 0$ **then**
4: **if** $((y_{j-1} - y_j) > \epsilon_V)$ or $((y_j - y_{j+1}) > \epsilon_V)$ **then**
5: y_j is Turning point
6: **end if**
7: **end if**

If a point P_j is selected to add to the compressed list, P_j is stored in the compressed point list and the information between P_i and P_j is deleted. P_i is replaced by P_j and the algorithm is repeated until no more data is received.

The input of Algorithm 3 which is the main algorithm, is , ϵ_A and the original signal $P = \{P_1, ...P_N\}$. The algorithm generates the output of compressed set indices $C = \{c_1, c_2, ...c_M\}$.

The reconstructed set $\hat{Y} = \{\hat{y_1}, \hat{y_2}, ..., \hat{y_N}\}$ can be calculated as follows:

Algorithm 3 Least Area Algorithm

INPUT: A sequence of $\{P_1, ..., P_N\}$ and ϵ_A
OUTPUT: A sequence of $\{c_1, ..., c_M\}$
$i \leftarrow 1$, $k \leftarrow 2$
$C \leftarrow C \bigcup \{i\}$ {compression set contains the first point}
while P_k exists **do**
  connect P_i to P_k
  calculate the $Area_{i,k}$ (using Algorithm 1)
  if $Area_{i,k} > \epsilon_A$ **then**
    $C \leftarrow C \bigcup \{k-1\}$
    $i \leftarrow k-1$ {replace the first point with P_{k-1}}
  else
    determine if P_k is a Critical Turning Point algorithm (using Algorithm 2)
    if P_k is a turning point **then**
      $C \leftarrow C \bigcup \{k\}$
      $i \leftarrow k$ {replace the first point with P_k}
    end if
    $k \leftarrow k+1$ {Move to the next processing point}
  end if
end while

$$\hat{y}_i = \begin{cases} y'_k & \text{if } i = c_k, \\ f(y'_k, y'_{k+1}) & \text{such that } c_k < i < c_{k+1}. \end{cases} \tag{3.3}$$

$f(y'_k, y'_{k+1})$ is linear interpolation between y'_k and y'_{k+1}. Therefore if y'_k and y'_{k+1} is connected by a line, each point on that line at equal time intervals between these two points represent the values of $\hat{y_n}$ in the reconstructed graph.

3.6 Complexity

Suppose N is the total number of points in a signal and M is the maximum number of points between two turning points.

Lemma 1. *The complexity of the Least Area algorithm is* $O(MN)$.

Proof. In the Algorithm 3, since there are N points in original sequence, the while loop in line 3 runs N times and has a complexity of $O(N)$.

In Algorithm 1, the area is calculated between each two significant turning points. Since there are M points between two turning points, it would be executed M times and the complexity of this algorithm is $O(M)$. This algorithm is being called in line 5 of Algorithm 3, therefore line 5 takes $O(M)$.

Algorithm 2 has a complexity of $O(1)$ and since all other lines in the Least Area algorithm also take $O(1)$, it can be concluded that the complexity of Algorithm 3 is $O(NM)$.

□

According to Lemma 1 the complexity of Algorithm 1 is $O(MN)$. In practice, the value of M is small ($M < 60$); therefore, the expected running time is $O(N)$ or linear.

Experimental Observation *The expected running time of Algorithm 3 is linear.*

The experimental values, such as in Table 3.5, shows that even for large values of ϵ_A which resulting in a larger M, the value of M is relatively small when compared to N. For example in Table 3.5, a sequence with $N = 15000$ points, M can only grow up to 100 which is a small number.

Test	ϵ_A	CR[a]	min M	max M	avg M
TA[b]	0.1	0.0484	3	176	22.8
	0.05	0.0884	3	79	12.8
	0.03	0.126	3	64	9.9
	0.02	0.1864	3	50	7.2
	0.01	0.33	2	16	5.0
	0.008	0.3952	2	14	4.5
	0.005	0.5336	2	11	3.8
TH[c]	0.05	0.05	4	172	22.4
	0.02	0.097	4	84	12.3
	0.008	0.17	3	36	7.8
	0.005	0.23	3	23	6.3
	0.001	0.6	2	8	3.9

[a]Compression Ratio
[b]AHA Database, N=2500 samples
[c]Holter Database, N=15000 samples

Table 3.5: Effect of ϵ_A on selected number of points

3.7 Storage Requirement

When processing the algorithm, no data is stored until one point is selected. To store the selected point, there is no need to have extra storage, because what is done is just deleting the points between that and the last selected point. Therefore, all we need is an storage to store N input points.

Since the time interval between two points are equal, each point is stored by two integer entities. One shows its index (instead of time value) and the other shows its voltage. To store each entity, an unsigned short integer is used. It needs only two

Bytes storage and can keep an integer up to $2^{16} - 1 = 65536$ that is enough for this technique. Therefore, the total storage needed is $2N$ Bytes.

The maximum number of points required to store as input, is the number of points between two critical turning point M plus one. The information of the point after critical turning point is needed otherwise the turning point can not be distinguished.

As the result amount of storage needed is $2(M + 1)$ which is linear order of M or $O(M)$.

3.8 Summary

In this chapter, the online ECG signal compression problem is defined. It is shown how a decision can be made as each point is read and as the decision is made the information about other points is deleted. This algorithm stores as much information as it needs. The major advantage of this algorithm is that the problem can be solved in linear time and can be used online.

Chapter 4

Modified CCSP method

In this chapter we review the Cardinality Constrained Shortest Path (CCSP) method which is a time domain signal compression scheme based on a rigorous mathematical model of the compression problem [29]. The CCSP algorithm requires all the data to be present prior to processing. It makes the algorithm dependant of the data duration and the complexity of the algorithm is easily become $O(MN^2)$. However CCSP generates high quality graphs in compare to the rest of time domain algorithms, its running time prevents it to be used in on-line applications. We modified the algorithm and investigated the ways the CCSP algorithm can be independent of the signal duration and reduce the execution time significantly while preserving the same signal quality which CCSP generates.

Section 4.1 is devoted to some definitions used in this chapter. In Section 4.2 we describe the CCSP algorithm. Section 4.3 is devoted to the modification of the CCSP algorithm. The MCCSP (Modified CCSP) algorithm is presented in Section 4.4. Complexity is discussed in Section 4.5 and 4.6 is the summary of this Chapter.

A crucial point in the development of the different compression schemes is the investigation of the computational complexity of the algorithm. The goal in this work was to achieve an execution time lower than CCSP.

4.1 Preliminaries

Definition of Arc: Let $Y = \{P_i, P_{i+1}, ..., P_{j-1}, P_j\}$. An arc is an ordered pair (i, j) such that $i, j \in Y$ and $i < j$.

Definition of Path $Path_{i,j,k}$: Let $Y = \{P_i, P_{i+1}, ..., P_{j-1}, P_j\}$. $Path_{i,j,k}$ is an ordered set $W = \{V_0, V_1, ...V_{k+1}\}$ such that $V_0 = P_i$, $V_{k+1} = P_j$ and $W \subseteq Y$.

Definition of Length of a $Path_{i,j,k}$: Length of a $Path_{i,j,k}$ is the number of arcs k, on the path.

For example, a path between P_1 and P_4 with the length of one, is represented by $Path_{1,4,1}$. $Path_{1,3,2}$ represents a path from P_1 to P_3 with both arc $(1, 2)$ and $(2, 3)$ in it.

Notation of Segment: Segment $\overline{y_i y_j}$ is the line connecting the arc (i, j).

Notation of $\hat{y}_k$: $\hat{y}_k$ is the the approximation value of y_k on the arc (i, j).

Since we use a linear interpolation to represent the value of segment $\overline{y_i y_j}$, $\hat{y}_k$ can be calculated as followings:

$$\hat{y}_k = \frac{k - i}{j - i}(y_j - y_i) + y_i \tag{4.1}$$

Definition of Approximated Distance of an arc (i, j):

The approximated distance of an arc (i, j) (ADA) is given by:

$$d_{i,j} = \sqrt{\sum_{k=i}^{j} (\hat{y}_k - y_k)^2} \tag{4.2}$$

Definition of Error $E(i, j, k)$:

$E(i, j, k)$ is the error of $Path_{i,j,k}$ which is the summation of approximated distance of all arcs along the path.

$Path_{i,j,k}$ has minimum error if the summation of approximated distance of all arcs (ADA) along the path is minimized.

4.2 CCSP algorithm

Suppose the original graph is given by $Y = \{y_1, y_2, ..., y_N\}$ at equal time intervals. Let the compressed set be $Y' = \{y'_1, y'_2, ..., y'_M\}$ and there exists a set $C = \{c_1, c_2, ..., c_M\}$: $1 \leq c_i \leq N$ and $c_i < c_{i+1}$ and $c_1 = 1, c_M = N$ such that $y'_i = y_{c_i}$ and the $Path_{1,N,M-1}$ has minimum error. The reconstructed set is a set $\hat{Y} = \{\hat{y_1}, \hat{y_2}, ..., \hat{y_N}\}$ [9].

4.2.1 The error of $Path_{i,j,k}$

The error in this algorithm is the summation of the error of all the paths within the signal. The path $Path_{i,j,k}$ in Figure 4.1 is chosen based on the number of points to be chosen in a path and the minimum path error[1] between points P_i and P_j.

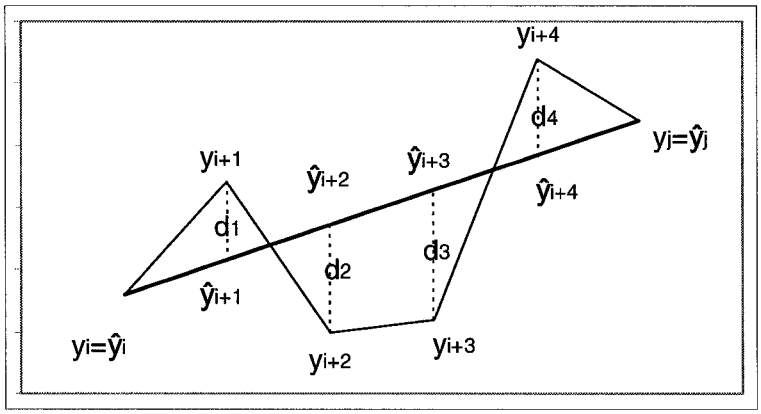
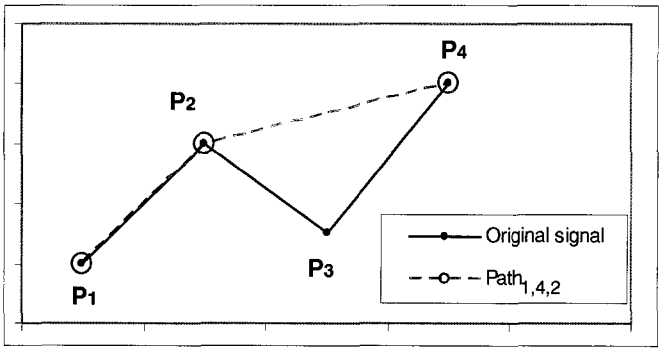


Figure 4.1: $Path_{i,j,1}$

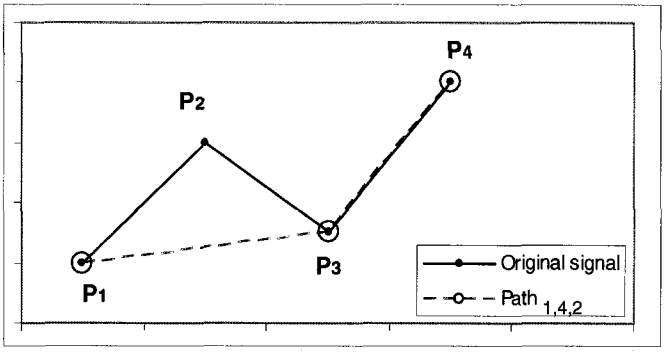
The CCSP algorithm starts from the first point and assumes the path with minimum error has a length of one. It calculates all paths from the first point to all other points of the length of one which are $Path_{1,1,1}, Path_{1,2,1}, ..., Path_{1,N,1}$. Then it

[1]Finding the path with minimum error is called the shortest path problem in the original algorithm [23]

computes a path of length two which are $Path_{1,3,2}, Path_{1,4,2}, ...Path_{1,N,2}$. There are different possibilities for a path with a length more than one. For example in Figure 4.2, $Path_{1,4,2}$ can be either the set $\{y_1, y_2, y_4\}$ or $\{y_1, y_3, y_4\}$.



(a) P_2 is on the path



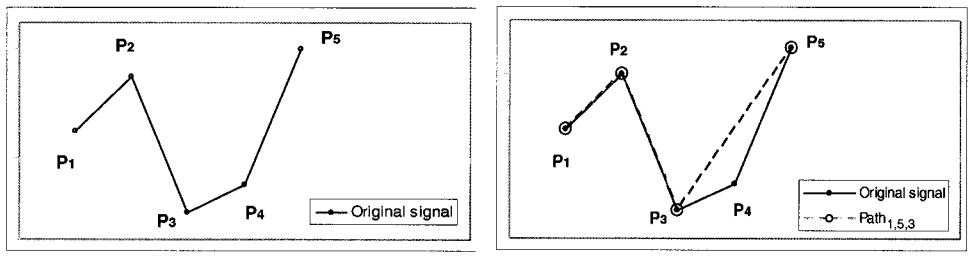
(b) P_3 is on the path

Figure 4.2: Different possibilities for $Path_{1,4,2}$

The algorithm first chooses P_2 and calculates the path error for $\{y_1, y_2, y_4\}$. Then it selects P_3 and calculates the path error for $\{y_1, y_3, y_4\}$. It compares these two errors and decides which path has the minimum error and stores the information of

this path.

CCSP computes the error of a path with a certain length by adding a new point each time and calculating all possible path errors until it processes all the points. Then it starts from the first point and increases the length of the path until it reaches a path of length $M-1$. An example for 5 points and path selection is presented in Figure 4.3. For each path, CCSP stores the last and second last point of the path and later uses this information to restore the compressed sequence.



(a) Original Graph

(b) P_2 and P_3 are on the path

(c) P_2 and P_4 are on the path

(d) P_3 and P_4 are on the path

Figure 4.3: Different possibilities for $Path_{1,5,3}$

The CCSP algorithm calculates errors based on previously stored information and therefore falls into the dynamic programming category. It also finds the optimum path from the first to the last point with minimum possible error. Its major difference with

other compression algorithms is in receiving a specific compression ratio as its input.

4.3 Modifications to the CCSP algorithm

In order to reduce the computational complexity of the CCSP and then prepare it for on-line applications, we have to compute all the arc distances in a careful manner. The original graph Y consists of $\frac{N(N-1)}{2}$ arcs. This means that we have $O(N^2)$ arcs for which we need to compute the distance. From Equation 4.2 it is easily seen that the expression for $d_{i,j}$ is a sum of $(j-i+1)$ terms, a number of order N. This means that straightforward computation of all these arc distances will result in an algorithm with complexity $O(N^3)$. Fortunately, this can be avoided by applying a careful strategy. The original graph can be broken into meaningful sections and each section can then be processed independently. Then all sections can be put back together. In each section, an error different from the CCSP error definition can be considered.

4.3.1 Error definition

Assume the graph G consists of N points, $G = \{y_1, y_2, ..., y_N\}$. Consider a part of the graph $G' = \{y_i, y_{i+1}, ..., y_j\}$. In equation 4.2 only y values are considered in calculating the error. The values of $\hat{y_n}$ will be easily calculated by the intersection of the segment $\overline{y_i y_j}$ and the line from the selected point parallel to y-axis.

In order to achieve a closer curve we incorporate the area generated between the original graph and arc (i, j). Each area is considered to be a positive value so as to keep the error growing over time. The area between two graphs can be calculated easily as explained in Section 3.2.2.

This is illustrated in Figure 4.4 where y_i is the first point selected in the graph G and and y_j is the last point.

$$TotalArea = a_1 + a_2 + a_3 \tag{4.3}$$

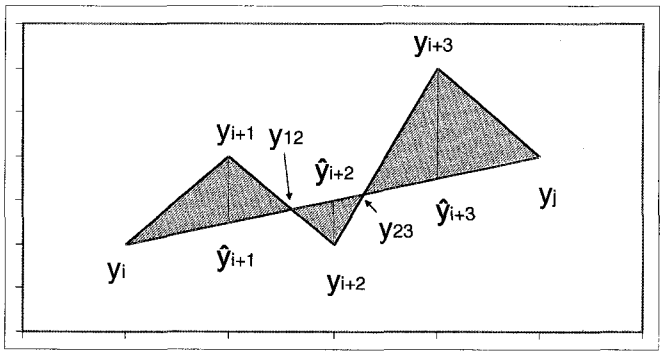


Figure 4.4: Area between segment $\overline{y_i y_j}$ and original graph

Area	Type
a_1	triangle $y_i y_{i+1} y_{12}$
a_2	triangle $y_{12} y_{i+2} y_{23}$
a_3	triangle $y_{23} y_{i+3} y_j$

Table 4.1: Different Areas generated in Figure 4.4

The areas in Figure 3.6(a) are shown in Table 4.1. y_{12} is the intersection point between two segments $\overline{y_{i+1} y_{i+2}}$ and $\overline{y_i y_j}$ and y_{23} is the intersection point between two segments $\overline{y_{i+2} y_{i+3}}$ and segment$\overline{y_i y_j}$

The segment $\overline{y_i y_j}$ may not intersect the original signal at any other points other than y_i and y_j (Figure 4.5). The generated areas are either triangle or trapezium. There are four areas in Figure 4.5:

$TotalArea = a_1 + a_2 + a_3 + a_4$ These area are shown in Table 4.2.

In MCCSP, we want to minimize the summation of the areas instead of arc distances, as we did for *Least Area* algorithm.

$$d_{ij} = Area_{i,j} \tag{4.4}$$

In this way the value of $d_{i,j}$ is different in this modified algorithm as can be seen in

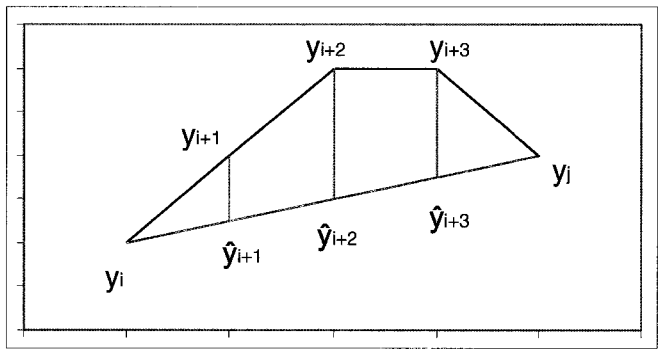


Figure 4.5: Area between segment $\overline{y_i y_j}$ and original graph

Area	Type
a_1	triangle $y_i y_{i+1} \hat{y_{i+1}}$
a_2	trapezium $y_{i+1} y_{i+2} \hat{y_{i+2}} \hat{y_{i+1}}$
a_3	trapezium $y_{i+2} y_{i+3} \hat{y_{i+3}} \hat{y_{i+2}}$
a_4	triangle $y_{i+3} y_j \hat{y_{i+3}}$

Table 4.2: Different Areas generated in Figure 4.5

Figure 4.6. The modified error measurement is exactly the error measurement which has been discussed in Section 3.4.1.

4.3.2 Input Point Limitation

CCSP has a complexity of $O(N^3)$ [10]. In order to speed up the running time of MCCSP, we process the input data in batches instead of the whole input stream. Experimental results in Table 4.1 show that if the number of points in each batch increases, the algorithm takes more time to select the compression set.

In any computer application, receiving each cycle of input data, processing it and generating the result takes a certain amount of time based on the available hardware and system configurations. In an on-line application when the next cycle of data is received, the output of the previous cycle must have been generated. Considering the

fact that receiving and printing the data always takes the majority of the total time, it has been assumed that the execution time of the algorithm is less than 2% of the data recording time[2]. With this assumption in every 100 seconds the algorithm can have at most 2 seconds execution time. Table 4.3 shows a test which has a time interval of 0.004 seconds. Usually a compression ratio of 30% is the maximum compression ratio which is intended to obtainable with compression algorithms. In Table 4.3, for 30 points in original graph and 20 points in compressed set, the execution time of the algorithm was 0.031 seconds, while the recording time was 30*0.004=0.12 seconds. Since the assumption of the algorithm allows the execution time to be less than 2%, Maximum execution time can be 0.02*0.12=0.024 seconds. The execution time in this case is less than the maximum execution time. It can be seen for the compression ratio of 30%, only a batch of 30 points has less than the allowed execution time.

From Table 4.3 it seems 30 is a good choice for input data. We have executed four different tests, Test1[3], Test2[4], Test3[5] and Test4[6], each with only 30 input points for several sections of the signal. Table 4.4 shows the result of these tests. Since all the execution time of these tests are within the maximum 2% of total time, we conclude that is a good choice for input data.

The algorithm processes the first few points to find out the frequency of the signal, calculates the execution time with 30 limited points and compression ratio of 0.33. If the execution time is less than 2% of the 30 input data, 30 is set as the point limit, otherwise it calculates the execution time for less than 30 times until it finds a good limitation number for points. This technique decreases the execution time efficiently.

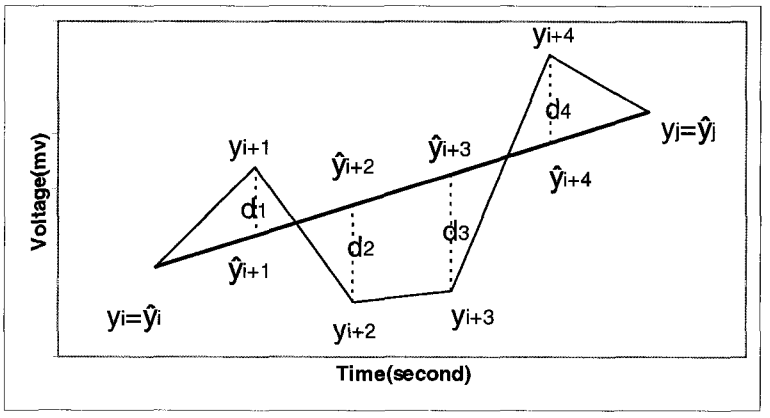
[2]If the receiving and printing time is improved, the algorithm can process more data
[3]PTB Diagnostic ECG Database
[4]Noise Stress Test Database
[5]Sudden Cardiac Death Holter Database
[6]PAF Prediction Challenge Database

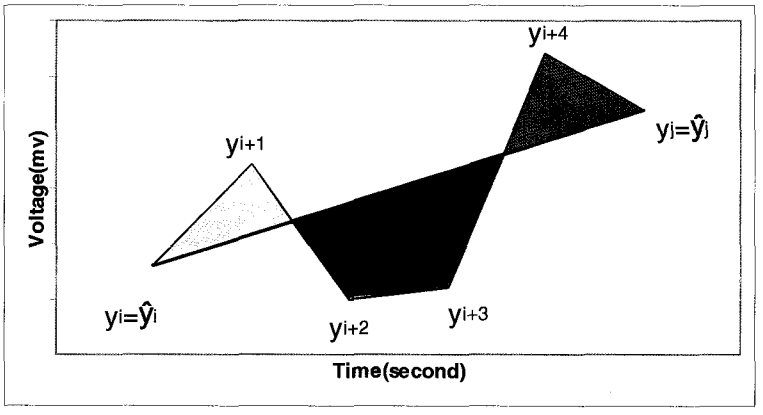
Number of limited points	M	Execution Time (Seconds)	Recording Time (Seconds)	Maximum Execution Time(Seconds)
30	5	0.015	30*0.004=0.12	0.12*0.02=0.024
	10	0.010		
	20	0.031		
40	5	0.031	40*0.004=0.16	0.16*0.02=0.032
	10	0.047		
	20	0.052		
	30	0.063		
50	5	0.031	50*0.004=0.2	0.2*0.02=0.04
	10	0.078		
	20	0.125		
	30	0.125		
60	5	0.079	60*0.004=0.24	0.24*0.02=0.048
	10	0.142		
	20	0.272		
	30	0.382		
70	5	0.086	70*0.004=0.28	0.28*0.02=0.056
	10	0.152		
	20	0.316		
	30	0.414		
80	5	0.107	80*0.004=0.32	0.32*0.02=0.064
	10	0.186		
	20	0.347		
	30	0.462		

Table 4.3: Effect of the number of points in execution time

Test	Time Interval(second)	Number of limited points	M	Execution Time(second)	Max Execution Time(seconds)
Test1	0.001	30	10	0.006	0.006
Test2	0.003	30	10	0.012	0.018
Test3	0.004	30	10	0.017	0.024
Test4	0.008	30	10	0.027	0.048

Table 4.4: Execution and Delay time for different test time intervals





(b) $d_{i,j}$ in MCCSP

Figure 4.6: Defining different $d_{i,j}$s in CCSP and MCCSP

4.4 MCCSP algorithm

A program has been developed to implement the MCCSP algorithm. Suppose there are N points in the original graph and the compressed set consists of M points. We search for a path from the first point to the last point such that its error is the minimum possible error. This path contains z points such that $1 \leq z \leq M$ and can be shown as $Path_{1,N,z-1}$ which must satisfy the following equation:

$$E(1, N, z-1) = min_{1 \leq k \leq M-1} E(1, N, k) \tag{4.5}$$

The program computes the path with minimum area between starting point and the first turning point. After compressing this part of the signal, the turning point becomes the new starting point and program continues. The information about previous points is deleted from the memory and the information about unprocessed points is being held until it is being used.

Consider the path $Path_{1,j,k}$. Suppose the point prior to P_j in this path which has been selected before, is point P_i. $Path_{1,j,k}$ consists of two paths: $Path_{1,i,k-1}$ and $Path_{i,j,1}$. $Path_{1,i,k-1}$ is the path with minimum area before adding P_j, otherwise another path with minimum area had been found and replaced $Path_{1,i,k-1}$ (Figure 4.7). Therefore $E(1, j, k) = E(1, i, k-1) + d_{i,j}$. To find the value of $d_{i,j}$, we use the technique described in Section 4.2.1. Algorithm 1 is used to calculate each area. Starting from P_i to P_j, all the paths generated by only two points are calculated. The path with minimum area is stored. Then the three-points-paths are calculated and the algorithm stores the shortest paths. Suppose $Path_{i,j,k-1}$ is the shortest path from P_i to P_{j-1} with k points in it. To calculate the $Path_{i,j,k}$, it is enough to calculate $d_{i,j}$ and add it to $Path_{1,i,k-1}$ value.

The process continues recursively such that the first path is always easy to calculate. $E(1, j, 1) = d_{1,j}$, which is the error of the path from the first point to any other points with only two points on it. After calculating all paths of length two, the paths of length three are calculated. The value of k is increased and the paths with

minimum area are calculated using:

$$E(i,j,k) = min\{E(i,l,k-1) + d_{lj} | l = \{k-1,,j-1\}\}$$

M is the number of points in the compressed sequence which the algorithm takes as one of its inputs. The original signal $P = \{P_1, ...P_N\}$ is the other input of the algorithm. The Algorithm 4 generates the output of compressed set $C = \{c_i, c_j, ...c_N\}$.

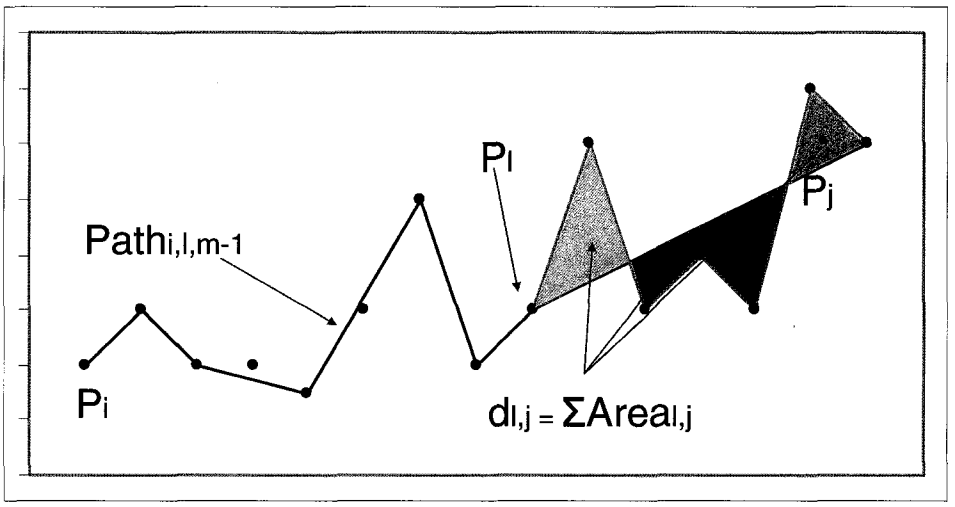
Path Finder algorithm (Algorithm 5) has been used to find the path with minimum area between two turning points in Algorithm 4. Algorithms first calculates the path from $P_{startIndex}$ to P_m with m points on it, plus a path from P_m to $P_{endIndex}$. $Path_{startIndex,endIndex,m+1} = \{P_{startIndex}, ..., P_m, P_{endIndex}\}$

Suppose there are other points between P_m and $P_{endIndex}$. The error of the path from the starting point to one of these points may be less than the previous case. Therefore Algorithms calculates the path from starting point and another point P_i, between P_m and $P_{endIndex}$, with m points on it plus the path from P_i to $P_{endIndex}$. $Path_{startIndex,endIndex,m+1} = Path_{startIndex,i,m} \bigcup \{P_{endIndex}\}$.

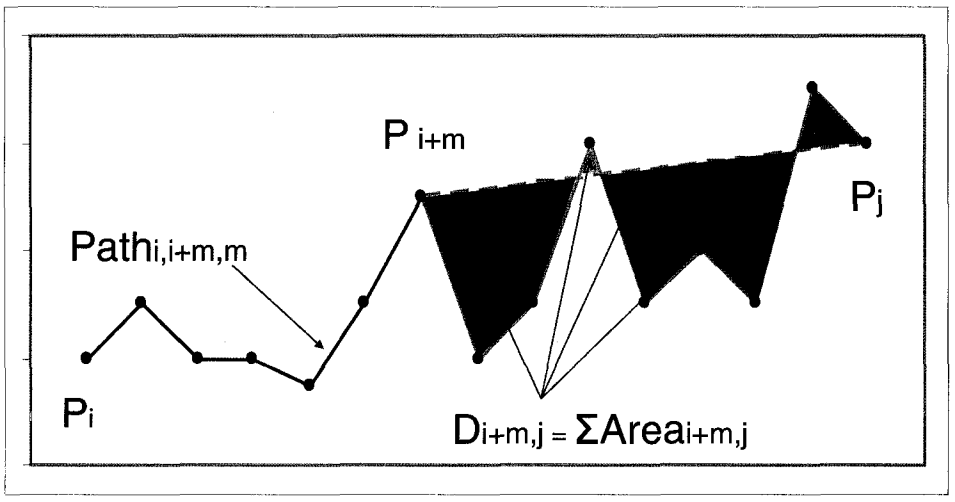
The Compression set C is used to keep the point index records. After the compression set $C = \{c_1, c_2, ..., c_N\}$ is completed, The values of $\hat{y}_i$ can be found easily.

$$\hat{y}_i = \begin{cases} y'_k & \text{if } i = c_k, \\ f(y'_k, y'_{k+1}) & \text{such that } c_k < i < c_{k+1}. \end{cases} \tag{4.6}$$

$f(y'_k, y'_{k+1})$ is linear interpolation between y'_k and y'_{k+1}. Therefore if y'_k and y'_{k+1} is connected by a line, each point on that line at equal time intervals between these two points represent the values of $\hat{y_n}$ in the reconstructed graph.



(a) calculate $E(i, l, m-1) + d_{l,j}$



(b) calculate $E(i, i+m, m) + d_{i+m,j}$

Figure 4.7: Adding y_j to the path from y_i to y_{i+m} with minimum area

Algorithm 4 MCCSP Algorithm

INPUT: A sequence of $\{P_1, ...P_N\}$ and M
OUTPUT: A sequence of $\{c_i, c_j, ...c_N\}$
$C \leftarrow \{1\}$
for $j \leftarrow 2$ to N **do**
  connect P_1 to P_j {generates a path}
  $E(1, j, 2) \leftarrow d_{1,j}$ {calculate the error of $Path_{1,j,2}$}
  $Path_{1,j,2} \leftarrow \{1, j\}$ {path of length 2}
end for
$k \leftarrow 2$ {Length of the optimal path}
$startIndex \leftarrow 1$
while $startIndex < N$ **do**
  $endIndex \leftarrow startIndex + 1$
  while $P_{endIndex}$ is not a *turning Point* **do**
    $endIndex \leftarrow endIndex + 1$
  end while
  $z \leftarrow \frac{endIndex - startIndex}{N} M$ {z (which is a portion of M) represents the number of points which can be selected between $P_{startIndex}$ and $P_{endIndex}$}
  if $z > 12$ **then**
    for $m \leftarrow 1$ to $z - 1$ **do**
      {$m + 1$ is the length of the path}
      for $j \leftarrow startIndex + m + 1$ to $endIndex$ **do**
        {j is the end point of the path}
        Call Algorithm 5(startIndex,j,m)
      end for
      if $E(startIndex, endIndex, m + 1) < E(startIndex, endIndex, k)$ **then**
        {If there is a longer path with less error than the optimal path, it replaces the optimal path}
        $k \leftarrow m + 1$
      end if
    end for
    $C \leftarrow C \bigcup Path_{P_{startIndex}, P_{endIndex}, m+1}$
  end if
  $startIndex \leftarrow endIndex$
end while

Algorithm 5 Path Finder Algorithm

1: INPUT: $startIndex$ and j and m
2: OUTPUT: $Path_{startIndex,j,m+1}$
3: $Path_{startIndex,j,m+1} \leftarrow \{P_{startIndex}, ..., P_{startIndex+m}, P_j\}$
4: $E(startIndex, j, m+1) \leftarrow E(startIndex, startIndex+m, m) + d_{m,j}$
5: {Error of $Path_{startIndex,j,m+1}$ is error of $Path_{startIndex,startIndex+m,m} = \{P_{startIndex}, ..., P_{startIndex+m}\}$ plus the error of the line connecting $P_{startIndex+m}$ to P_j}
6: **for** $i \leftarrow startIndex + m + 1$ to $j - 1$ **do**
7: **if** $E(startIndex, i, m) + d_{i,j} < E(startIndex, j, m+1)$ **then**
8: {Smaller error means $Path_{startIndex,j,m+1}$ is the path with smaller area}
9: $E(startIndex, j, m+1) \leftarrow E(startIndex, i, m) + d_{i,j}$
10: $Path_{startIndex,j,m+1} \leftarrow \{P_{startIndex}, ..., P_i, P_j\}$
11: **end if**
12: **end for**

4.5 Complexity

Suppose there are at most Q points between each two *turning points*. N is the total number of points in original graph and M is the number of points in compressed set.

Lemma 2. *The complexity of Algorithm 4 is* $O(N + MQ^2)$.

Proof. Since there are a total on N points in original graph, the **for** loop from line 4 to 8 takes $O(N)$. The while loops in line 11 and line 13 run N times in total. The reason is that the first time line 11 runs, it starts from $P_{startIndex}$, but the next time $P_{startIndex}$ has changed to a turning point instead of $P_{startIndex+1}$. Line 12 takes $O(1)$, and line 16 takes $O(1)$. The **for** loop in line 18 takes $O(z)$; since z is a fraction of M, and summation of all z is M; therefore **for** loop in line 18 takes $O(M)$ in total. Line 20 takes $O(Q)$. The **for** loop in Algorithm 5 also takes $O(Q)$ and lines 24 to 26 run in $O(1)$. Therefore the total running time for lines 18 to 28 is $O(MQ^2)$. As a result the total running time for Algorithm 4 is $O(N + MQ^2)$. □

4.6 Summary

In this chapter, the on-line ECG signal compression problem is studied from a different perspective. It is shown how a graph can be compressed without knowing any threshold, just by having the compression ratio. The strongest point of this algorithm is that the execution time of the algorithms has been improved significantly.

Chapter 5

Experiments

In this chapter the performances of the compression algorithms LA and MCCSP are presented. We show that the objective of this study which is accuracy and on-line capabilities have been fulfilled.

The MIT-BIH Arrhythmia database [19] was used to evaluate our algorithms and compare them with other known compression methods. The data sets have been chosen such that they certain different frequencies and durations. Not only are normal sinus rhythms tested, but also unusual and rare cases have also been studied. In order to test the dependency of the algorithms on the test data.

5.1 Compression Ratio

Compression by time domain methods are generally accomplished by keeping a set of significant signal samples. A logical way of measuring how well such compression methods compress a given signal, is to compare the number of retained samples in the compressed signal to the number of original signal samples. This is called the *compression ratio*, defined as the number of samples in the compressed signal per original signal sample and therefore its value is between 0 and 1. This number is one if and only if all the original samples are in the compressed set.

5.2 Measures of Performance

The performance of a compression algorithm can be measured in a number of different ways such as the distortion between the original and the reconstructed signal, the amount of compression, the complexity of the algorithm, the execution time on a given machine and the visual similarity between the original and the reconstructed signal. We roughly measured the performance of our compression algorithms by considering all of these measures.

5.2.1 Distortion Measures

Since ECG signals are generally are compressed using lossy compression algorithms, it is necessary to quantify the *distortion*, which is difference between the original and the reconstructed signals. Traditionally, two different distortion measures have been applied in ECG compression algorithms: The *maximum error* and the *summation of squared errors*. The maximum error is given by

$$maximum - error = max_{1 \leq n \leq N} |y_n - \hat{y_n}| \tag{5.1}$$

and the summation of squared errors by:

$$D = \Sigma_{n=1}^{N} (y_n - \hat{y_n})^2 \tag{5.2}$$

where y_n and $\hat{y_n}$ are the originals and the reconstructed signal, respectively, and N is the number of samples.

In order to be able to do a comparison between the performance of the different algorithms, we have applied two separate error measures, one based on the maximum error as given equation 5.1 and one based on the sum of squared errors, namely the commonly used *Percentage Root-mean-square Difference* (PRD) defined as [23]:

$$PRD = \sqrt{\frac{\sum_{n=1}^{N} (y_n - \hat{y}_n)^2}{\sum_{n=1}^{N} (y_n - \bar{y}_n)^2}} \times 100\% \tag{5.3}$$

where $\bar{y_n}$ is the mean value of the original signal. The smaller the PRD, the closer the reconstructed signal is to the original.

5.2.2 Visual Similarity

Recall that the purpose of an ECG compression algorithm is to achieve a maximum degree of compression while distorting the signal as little as possible. The quality required from the reconstructed signal is dependent on the specific application. For this reason, it is hard to give a general guideline as to what the maximum allowable distortion should be tolerated is, in order to obtain a reconstructed signal with the diagnostic information contained. This has to be decided by visual inspection of the signal.

However, there are some artifacts which are generally undesirable in the reconstructed ECG signal. Areas of the signal corresponding to the same electrical activities in the heart, should not be distorted in different ways between different periods of the signal. For example, the P-wave must not contain one artifact due to compression at one spot of the signal and a different at a another spot. This applies to ECG signals of periodic character, such as Normal Sinus Rhythm (NSR). For rhythms other than NSR, with less periodical characteristics, an important issue is that areas of the signal where the rhythm suddenly changes should not be distorted. These areas are of vital diagnostic importance as they can be very useful in deciding what mechanism that lead to a change of rhythm, and it is desirable to be able to reconstruct these segments with high fidelity.

Some distortion measures are also discussed which correlated to diagnostic information in the signal and are based on comparing the PQRST complex features of the original ECG signal and the reconstructed one, such as the PR wave duration and the QT duration. However this distortion measure limits the application of the compression algorithm to Normal Sinus Rhythm, it enables us to have a quantitative measurement of visual similarities.

5.2.3 Execution Time

By execution time, we mean the wall clock time that an algorithm takes to run, measured on a specific system using specific hardware and software configurations. This is often referred to as a benchmark. Benchmarks are dependent on the specific machine, compiler, input values and programming environment. Nevertheless, they tell us something about the fulfillment of on-line requirements, i.e., the ability of the algorithm to analyze the signal as it comes in. This is an important aspect of a compression algorithm.

One important input of some of the compression algorithms is threshold. The threshold has different definitions in each algorithm such as *Area threshold* or *Voltage threshold* and its value is dependant to the signal rhythm. A user must consult the specific signal to assign the threshold for the algorithm. Regardless of the method, the small thresholds will result in higher compression ratio and more accurate compressed signal.

We have compared our algorithms against the well-known FAN, CCSP, AZTEC, TP and CORTES algorithms in all four areas discussed above, i.e., PRD, maximum error, visual similarities and execution time. Although TP and CCSP generate the best results in terms of PRD, maximum error and visual similarities, CCSP has a high run time and TP can not achieve a compression ratio less than a certain amount[1]. FAN, AZTEC and CORTES have been chosen because of their linear run time and acceptable results.

5.3 The Experimental Setup

All of the algorithms implemented for our tests were implemented using Java, and executed on a Pentium 4 with a 3.2 GHz CPU and 2G of RAM.

[1]TP has a compression ratio of 0.5

The MIT-BIH Arrhythmia database [19] was used to evaluate the proposed algorithms and compare them with other known compression methods. Table 5.1 shows the test signals which were used in our experiments. More information about these test signals can be found in Appendix A.

Test	Database Name	Frequency(Hz)	Number of samples
T1	CU ventricular	250	125
T2	CU ventricular	250	250
T3	BIDMC	250	280
T4	BIDMC	250	350
T5	Arrhythmia	333	3600
T6	AHA	250	2500
T7	ANSI/AAMI EC13 Waveforms	667	7000
T8	PAF Prediction	125	1280
T9	PTB Diagnostic	1000	10000
T10	Sudden Cardiac Death Holter	250	15000

Table 5.1: Test signal characteristics

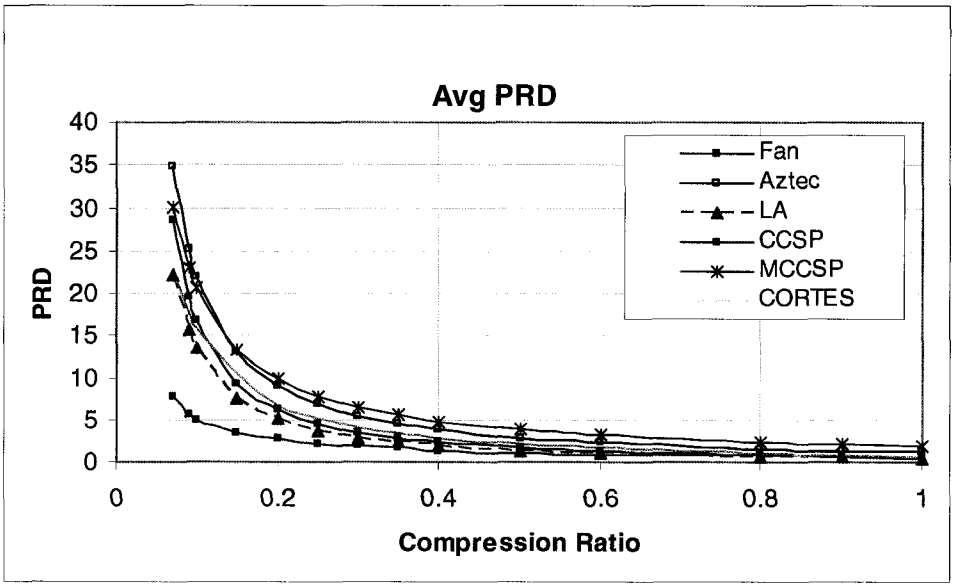
5.4 Evaluation Based On the PRD Measure

Four different test samples, T1, T2, T3 and T4 were used to compare each compression method with respected to their PRD. The average, minimum, maximum and median values of PRD versus compression ratio have been presented in Figures 5.1 and 5.2.

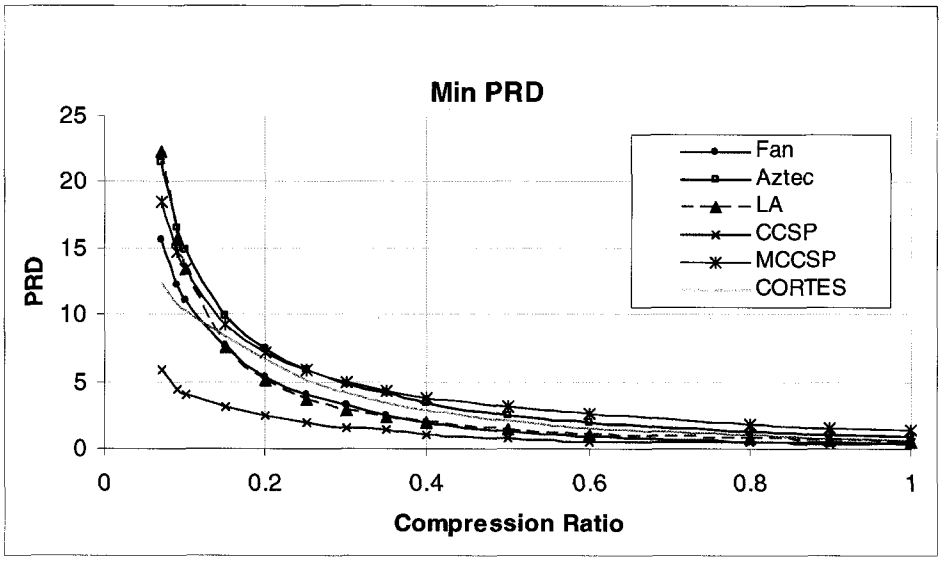
Analyzing the plots, we notice that CCSP has the best outcome. It is due to the fact that it finds a graph containing the points with the minimum possible error. By definition, it has the minimum PRD among other methods.

The LA method generally performs better than FAN and AZTEC algorithms especially when compression ratio is less than 30%. As it can be seen in Figure 5.3, the low frequency section of graph is the major source of the PRD of LA and high frequency points do not have a significant rule in calculating the PRD.

FAN has an intermediate PRD between all the methods. Figure 5.4 shows that

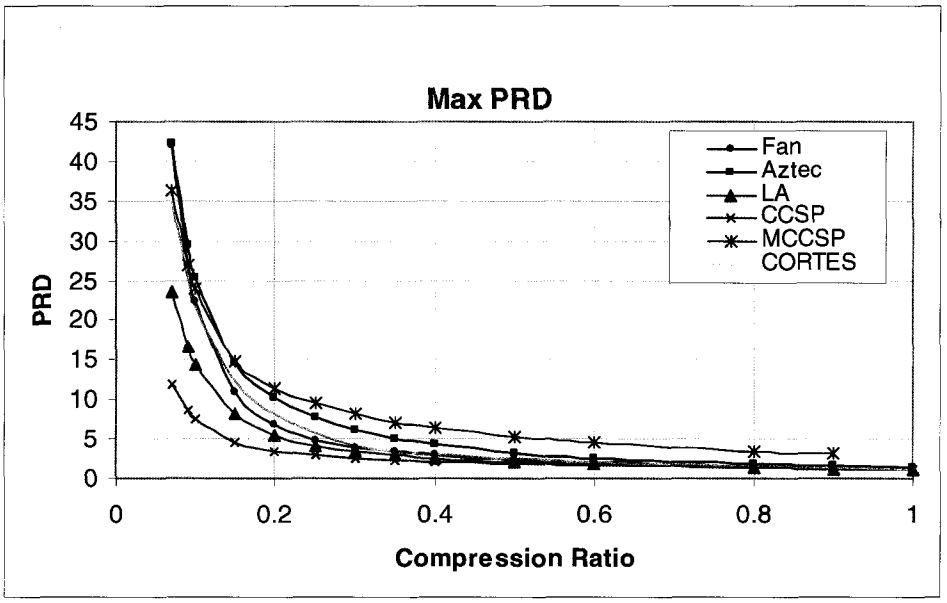


(a) Average PRD

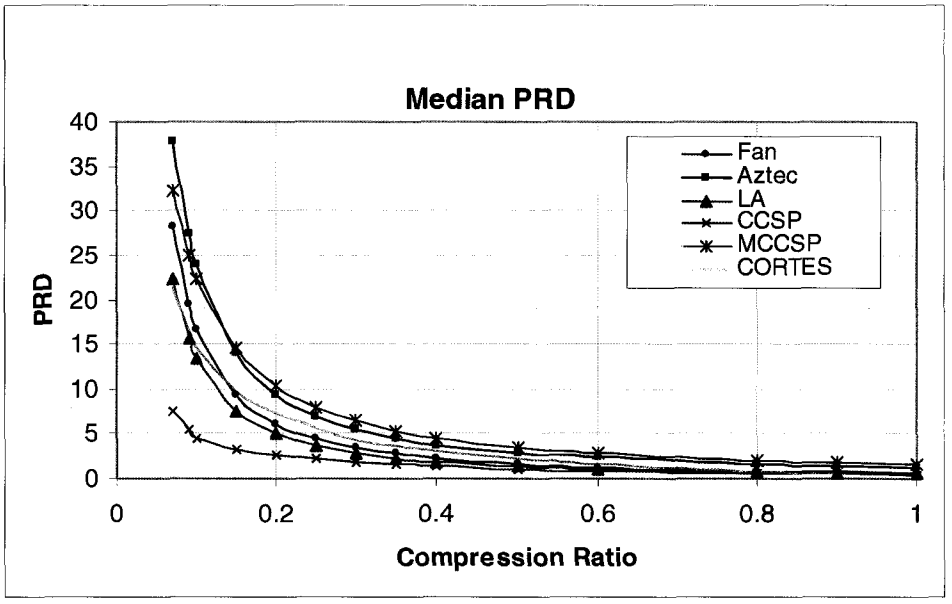


(b) Minimum PRD

Figure 5.1: Average and minimum PRD versus compression ratio



(a) Maximum PRD



(b) Median PRD

Figure 5.2: Maximum and Median PRD versus compression ratio

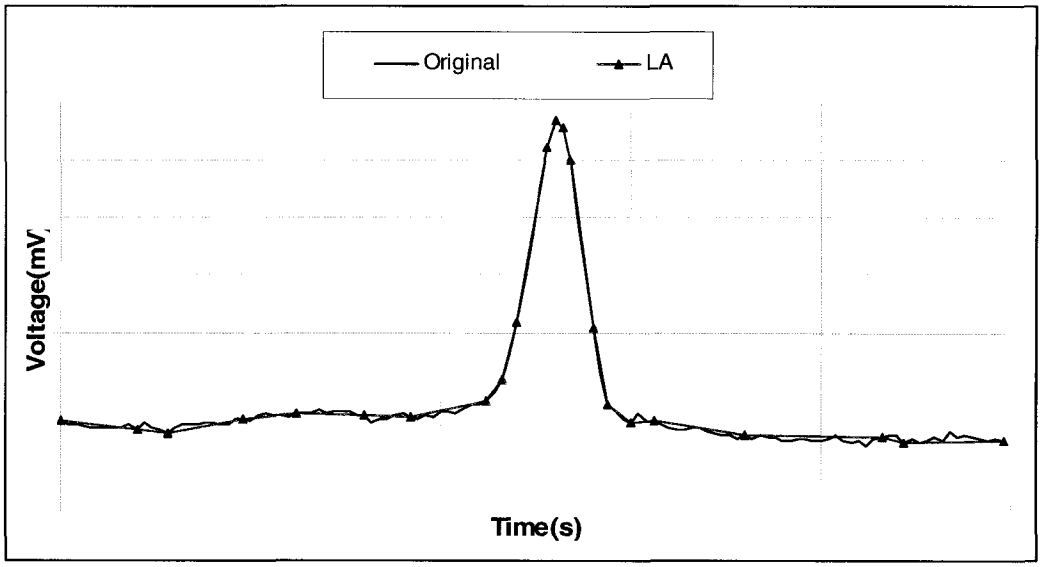


Figure 5.3: LA PRD

FAN has selected very good points in high frequency area, but in the rest of the graph the original and reconstructed points are different and they make FAN's PRD to become larger than LA.

MCCSP has the largest PRD among other methods. As can be seen in Figure 5.5 in section A of the graph, there are some points which have different constructed voltage values than in the original graph. In section B of the same graph, although the turning points have been chosen correctly and the graph looks similar to the original graph the reconstructed point values are very different than the original points. The reconstructed signal in low frequency parts, i.e. section C, is nearly identical to the original signal with a few different points. The combination of differences in section A, B and C are the cause of the larger PRD in the MCCSP.

The PRD of LA is outperformed by the CCSP in terms of PRD for all compression ratios of all the test signals. Although it has almost twice the PRD that CCSP has, its PRD is half of the the PRD of FAN and almost a quarter of the PRD of AZTEC.

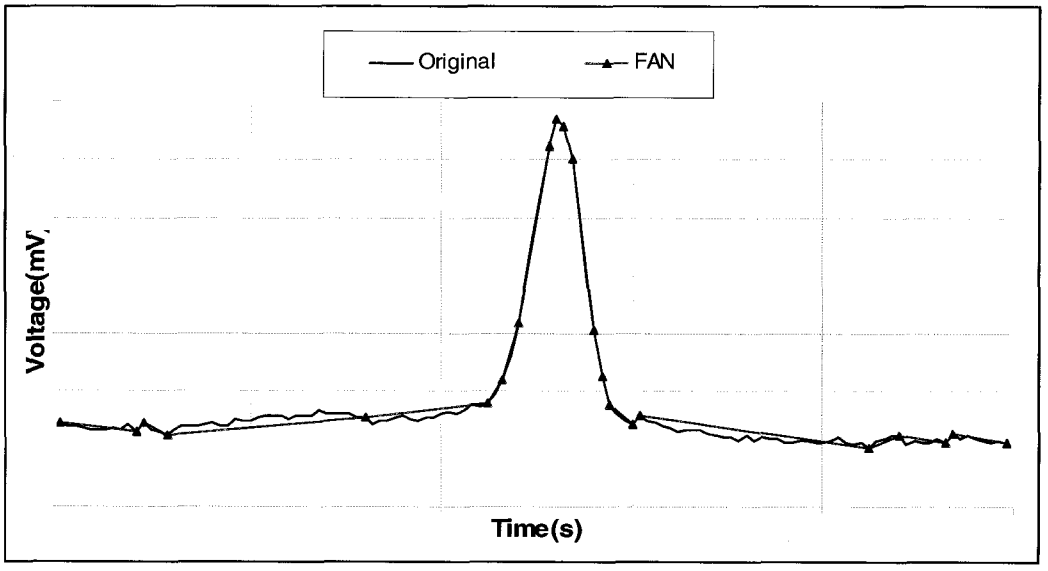


Figure 5.4: FAN PRD

Its PRD usually has the same PRD of FAN in compression ratios of 0.3. CORTES is slightly better than AZTEC and closer to FAN. It is because of compressed signal improvement in high frequency sections.

Table 5.2 shows the PRD values for compression ratio of 0.15 from Figure 5.1. All methods have been evaluated against CCSP and FAN in this table. PRD of LA is 2.18 times more than CCSP and 0.82 of FAN.

In order to test longer duration, three other data bases T5, T6 and T7 were used and the results are shown in Figures 5.6 and 5.7. These graphs show that LA has the least PRD. The difference between LA and other methods is bigger in smaller compression ratios, for example in compression ratio of 0.15, PRD of LA is 0.7, while for FAN is 0.9, for CORTES 0.95 and for AZTEC is 0.98. Performance of CORTES is very similar to AZTEC as in low compression ratios, since very few points are selected for plateaus.

Usually in compressing ECG signal, a compression ratio of less than 0.5 is required.

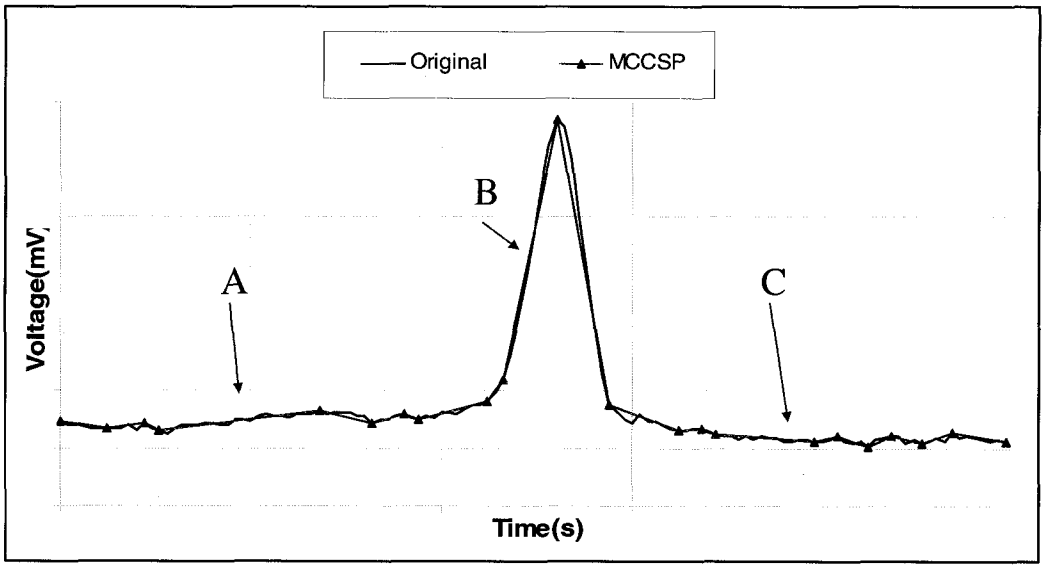
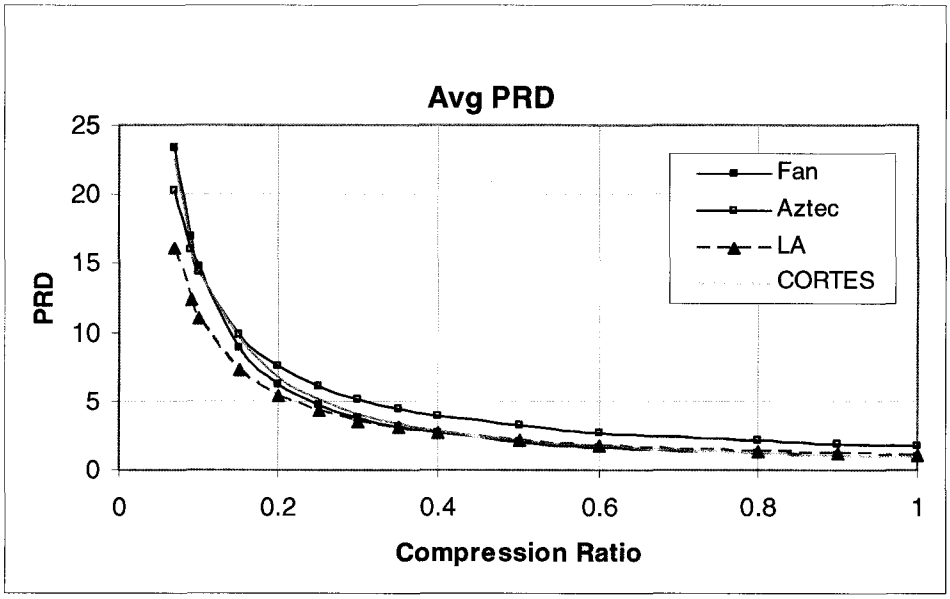
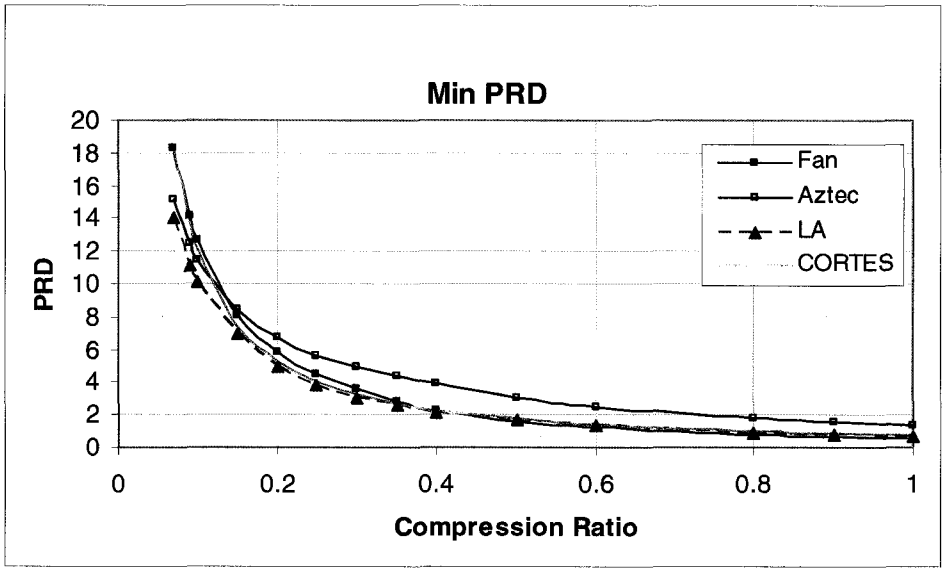


Figure 5.5: Effect of reconstructed points on the PRD of MCCSP

TP has a constant compression ratio around 0.5 for all the test samples. Even when the tests were repeated with the output generated by the algorithm as the input, the compression ratio did not improve more than 10%, therefore, the result of the first iteration of the algorithm was kept. The PRD for all algorithms are given in Table 5.3. Even for the compression ratio of 50%, CCSP, LA and Fan have better PRDs than TP. The reason is TP only selects the turning points and in some part of the signal like QRS complex, even small distortion from the original graph can make the PRD to increase.

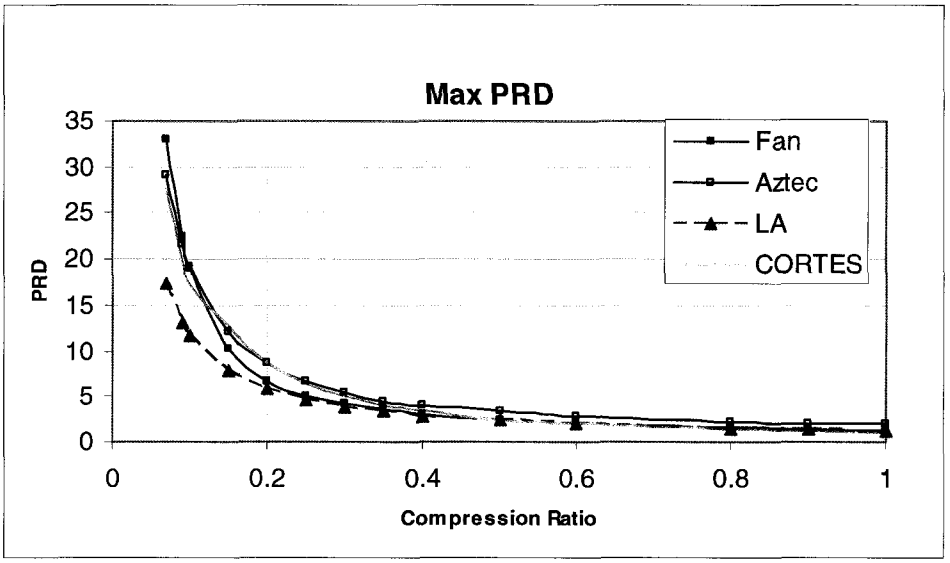


(a) Average PRD

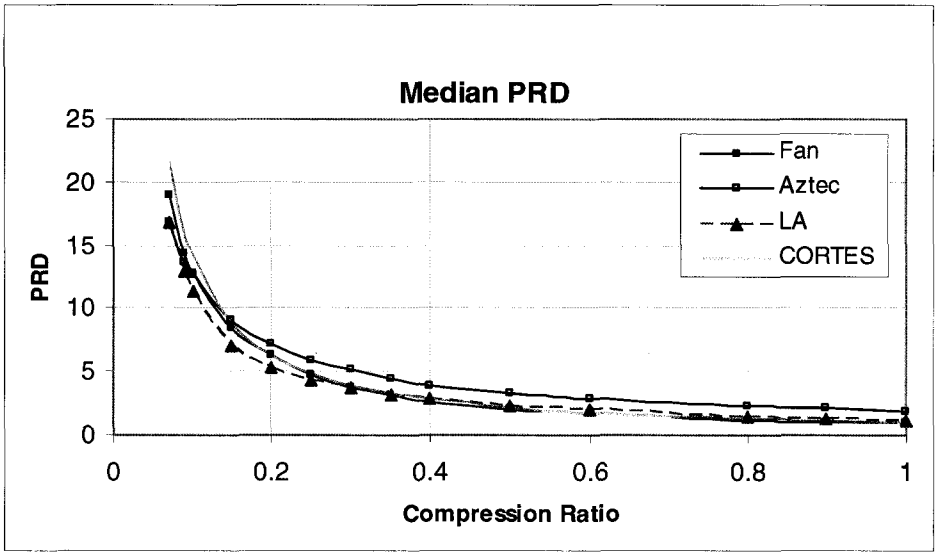


(b) Minimum PRD

Figure 5.6: Average and minimum PRD versus compression ratio



(a) Maximum PRD



(b) Median PRD

Figure 5.7: Maximum and median PRD versus compression ratio

Method	PRD	Relative PRD to CCSP	Relative PRD to FAN
Fan	9.26	2.67	1
AZTEC	13.12	3.78	1.42
CORTES	10.55	3.04	1.14
LA	7.59	2.18	0.82
CCSP	3.47	1	0.37
MCCSP	13.37	3.85	1.44

Table 5.2: PRD values for CR(Compression Ratio)=0.15 from Figure 5.1(a)

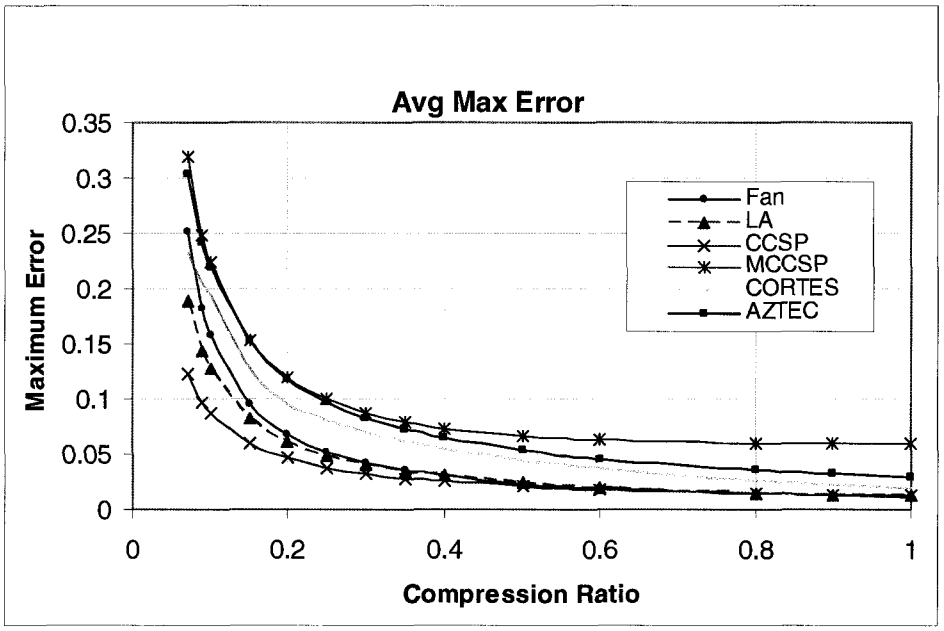
Algorithm	Average PRD
TP	2.04
Fan	1.75
AZTEC	2.87
CORTES	2.125
LA	1.39
CCSP	1.13
MCCSP	3.82

Table 5.3: Average PRD for all algorithms for CR=50%

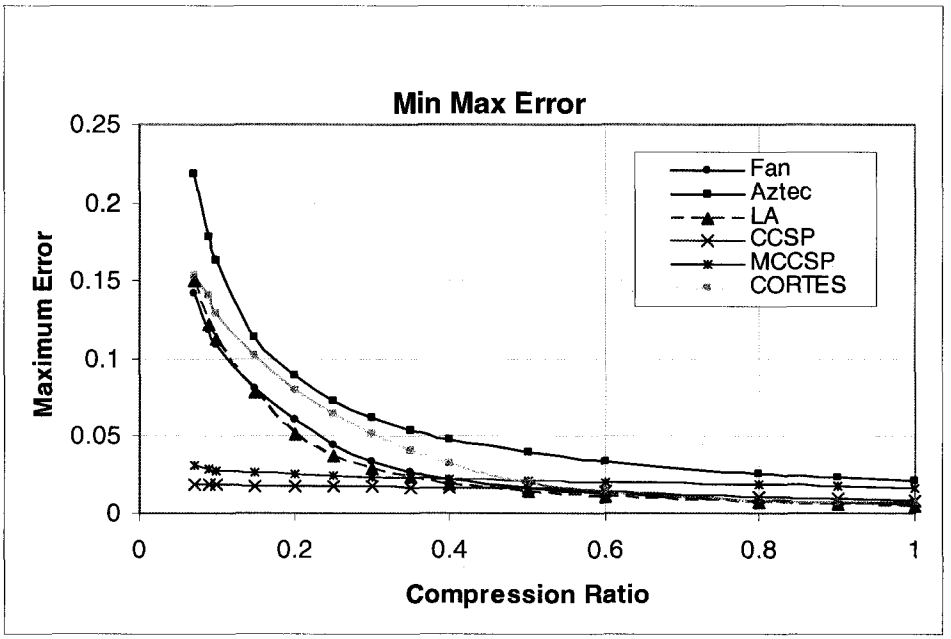
5.5 Evaluation Based On the Maximum Error

The maximum error was measured for test T1,T2, T3 and T4. The average maximum error (AME), maximum, minimum and median is shown in Figures 5.8 and 5.9.

These Figures show maximum errors as a function of compression ratio for different methods including CCSP and MCCSP. For each signal the experiments show that CCSP has the least amount of error in most compression ratios. In second place, LA has a very considerable low maximum error as described for Figure 5.3 in Section 5.4. AZTEC has the second highest maximum error as it considers the average voltage value and not selecting enough points for a specific time. All of the points which are approximated by the average line, have a distance with it and since all the signal is compressed by the average lines, the summation of the distances cause the PRD to be the largest value among all methods. Since CORTES selects the high frequency points

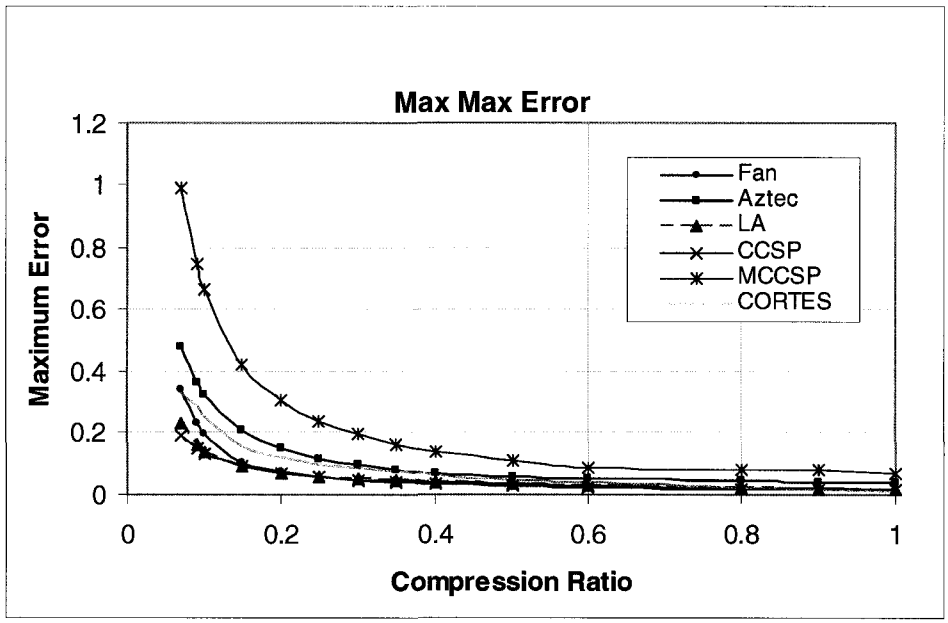


(a) Average Maximum Error

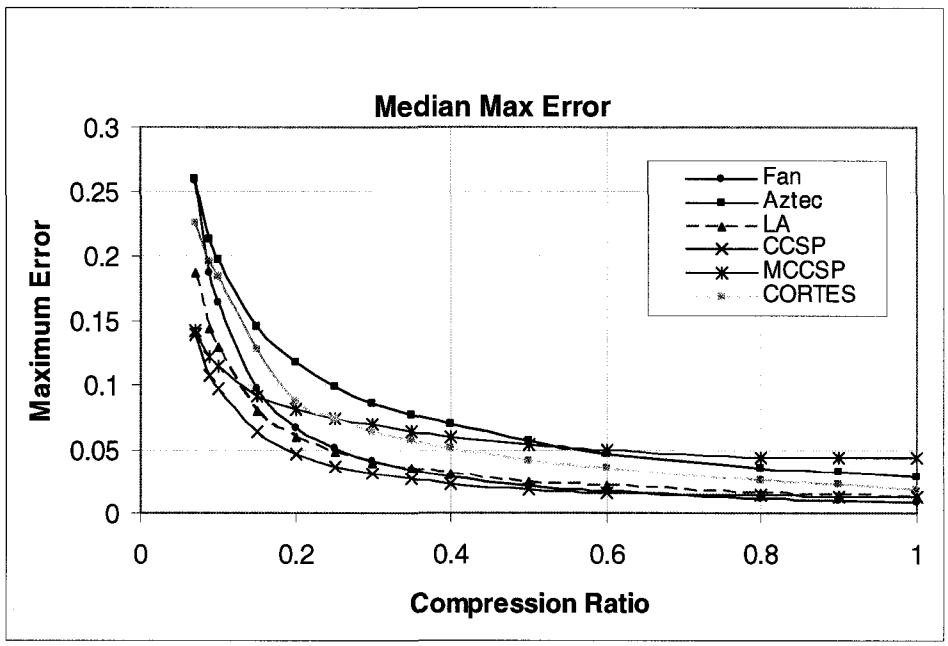


(b) Minimum Maximum Error

Figure 5.8: Average and minimum Maximum Error versus Compression Ratio



(a) Maximum Maximum Error



(b) Median Maximum Error

Figure 5.9: Maximum and median Maximum Error versus Compression Ratio

better (by using TP algorithm) and only approximate the low frequency section with average lines, its PRD is slightly better than AZTEC. The achievement of FAN is between LA and CORTES, since it selects the high frequency points very good as it is shown if Figure 5.4.

The behavior pattern of minimum value of the maximum error (MME) is different in CCSP and MCCSP with other methods. The reason is these two methods are minimizing the error between original and reconstructed graphs. Therefore their MME are always smaller than the rest of the methods. In the other side, as they works to minimize the maximum error, it is expected they keep the error of each point in a minimum range. The other methods works on threshold; therefore, the error of each point vary freely in the threshold range.

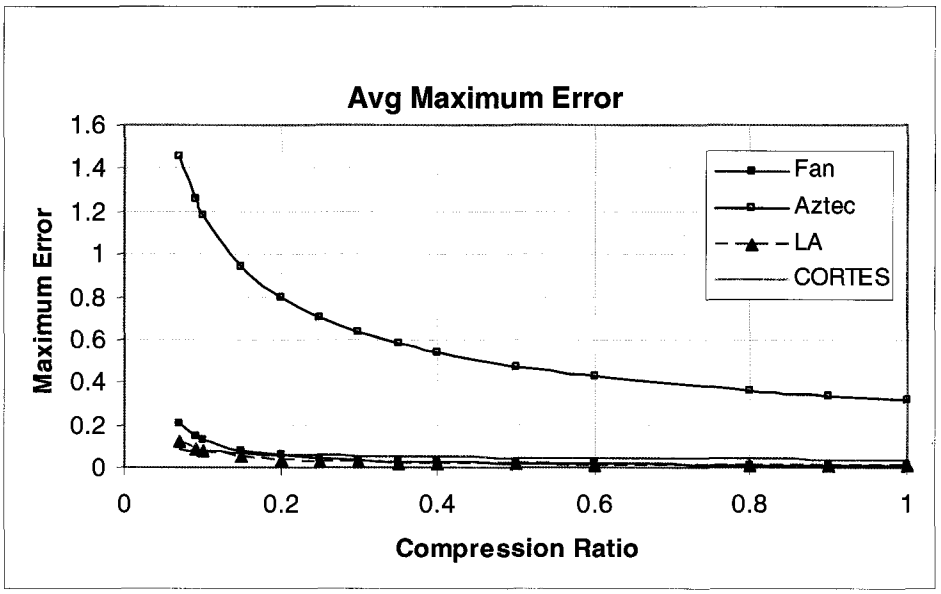
MME is always the minimum possible of maximum error value, because if it is not minimum, the algorithm is selecting the path which has the minimum error. Therefore the MME is almost a linear line with different behavior than the other algorithms.

MCCSP has the worst maximum errors due to its high frequency areas with low number of picked points as it was discussed for Figure 5.5. Therefore the difference between voltage values of the reconstructed and original points is higher than the other methods. In other parts of the signal, this method is generally performing better than AZTEC.

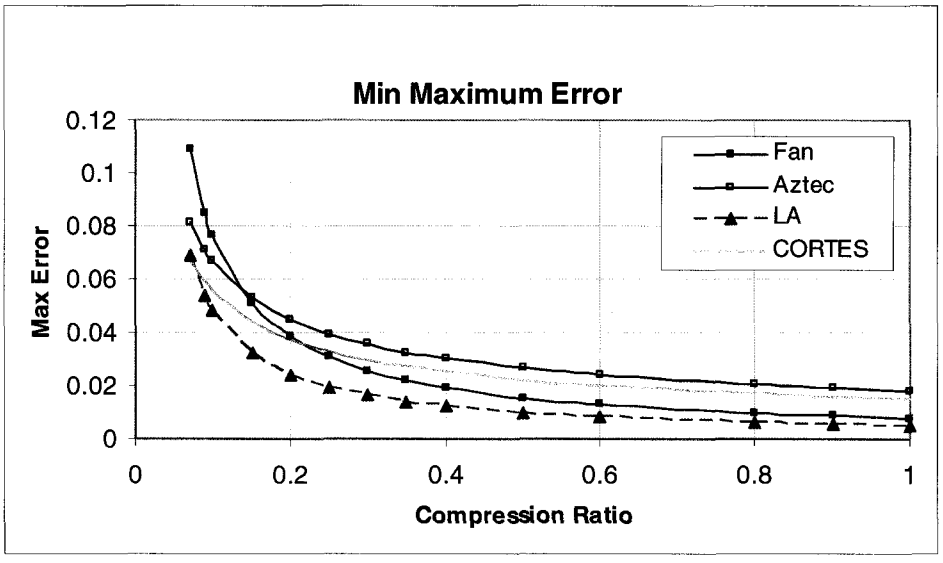
Table 5.4 shows the average maximum error for compression ratio of 0.15. The AME value for AZTEC and MCCSP is almost the same. For AZTEC the step like plateaus in low frequency areas cause the PRD to increase. In MCCSP the difference between reconstructed and original point values in high frequency area makes the PRD to grow. As far as CORTES is choosing all the turning points in QRS complex, it has lower AME.

To have a better comparison between FAN, AZTEC, CORTES and LA, they have been test with longer signals and the results can be found in Figures 5.10 and 5.11.

AZTEC has the highest maximum error between the other methods in Figure 5.10.

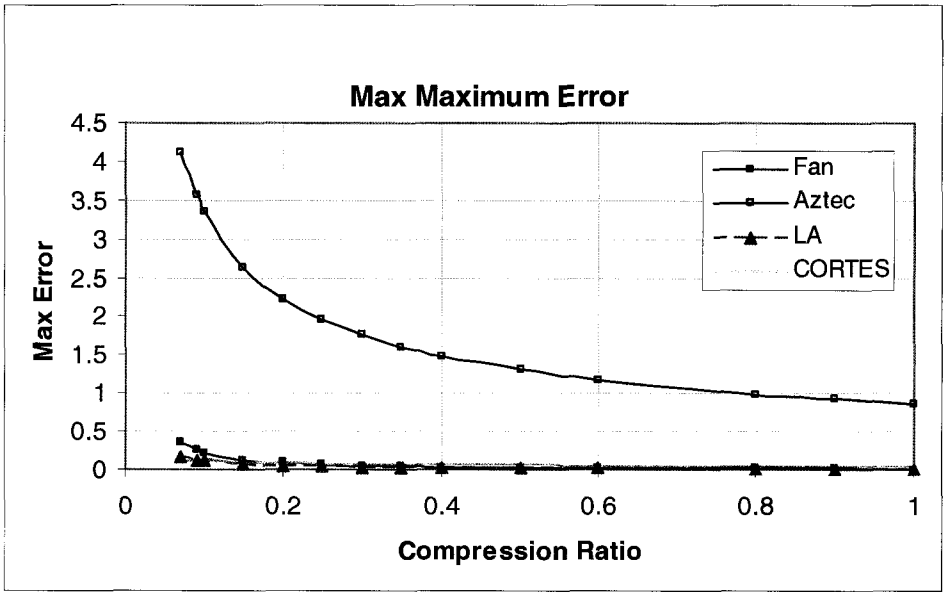


(a) Average Maximum Error

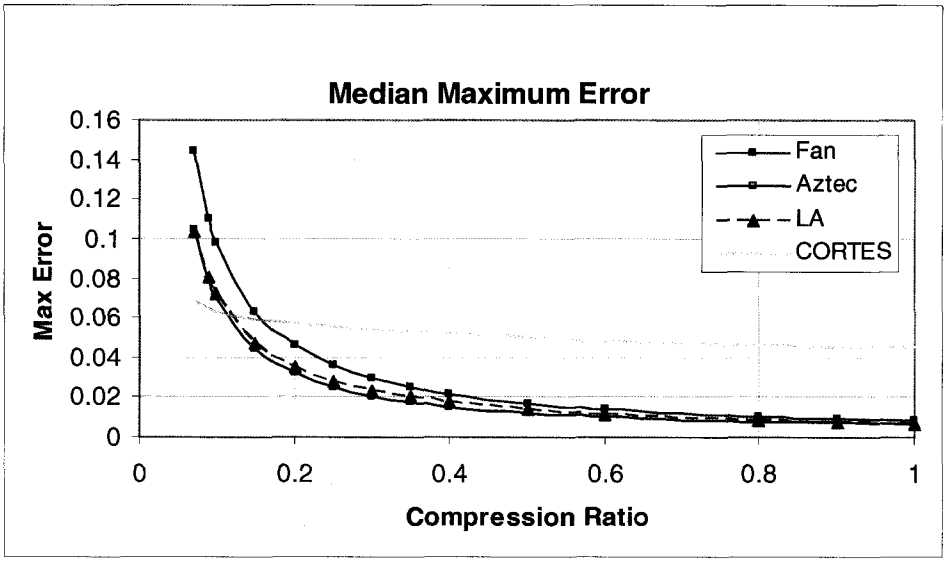


(b) Minimum Maximum Error

Figure 5.10: Average and minimum maximum error versus Compression Ratio



(a) Maximum Maximum Error



(b) Median Maximum Error

Figure 5.11: Maximum and median maximum error versus Compression Ratio

Method	AME	Relative AME to CCSP	Relative AME to FAN
Fan	0.095	1.61	1
AZTEC	0.153	2.60	1.61
CORTES	0.127	2.15	1.34
LA	0.083	1.40	0.87
CCSP	0.059	1	0.62
MCCSP	0.153	2.59	1.61

Table 5.4: Average Maximum Error(AME) values for Compression Ratio(CR)=0.15

Figure 5.12 shows that how AZTEC considers the average voltage value for a specific time. All of the points which are approximated by the average line, have a distance with it and since all the signal is compressed by the average lines, the summation of the distances cause the PRD to be the largest value among all methods.

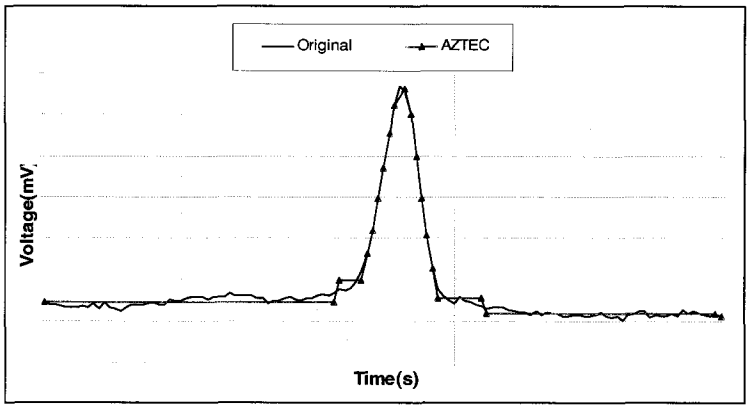


Figure 5.12: Effect of reconstructed points on the PRD of AZTEC

These Figures clearly show that LA performance is better than the other methods discussed earlier. However LA is achieving better results up to 30% and after that the error is small enough to be neglected.

Table 5.5 shows the average maximum error for all algorithms for compression ratio of 50%. The maximum error usually occurs in high frequency sections of the graph where the difference between y values of the original and reconstructed signal

are larger. Since TP selects the turning points only, its AME is the second highest.

Algorithm	AME
Fan	0.019
CCSP	0.021
LA	0.024
CORTES	0.044
AZTEC	0.053
TP	0.056
MCCSP	0.066

Table 5.5: Average maximum Error for all algorithms for compression ratio of 50%

5.6 Evaluation Based on Visual Inspection

The performance evaluation of the methods is accompanied with a visual inspection of the reconstructed signals. This is to show how similar the result of these algorithms are to original signal.

In Section 5.6.1 we studied the general performance of a long stream of data with two very low compression ratios 5% and 10%. This was to give an idea how low the compression ratio could be.

In Section 5.6.2 we examined a very specific part of the signal with two different compression ratios 10% and 17% and analyzed a couple of the important medical aspects of the signal which could have been measured from each graph mathematically. This section clarified the importance of visual inspection.

5.6.1 General Performance

The purpose of compression techniques are to reduce the amount of data while preserving the signal behavior. In order to save enough detail of the signal there is a limit for compression ratio regardless of the compression technique. For example, the points P, Q, R, S, T and U must be selected.

5.6.1.1 T6 - compression ratio of 5%

For the overall performance, a 10 second duration of T6 is selected with a compression ratio of 5% in Figures 5.13 to 5.16.

Figure 5.16 shows that for a compression ratio of 5%, LA generates an acceptable graph which is similar to the original signal than other methods. LA is more accurate in terms of picking the Q, R, S and T points as it has been observed in graphs 5.13 to 5.16. MCCSP shows more details in compare to other methods. Its performance is more detail oriented than LA. The reason is within an specific section of the graph, MCCSP finds the minimum possible area, while LA finds the area which is less than its threshold.

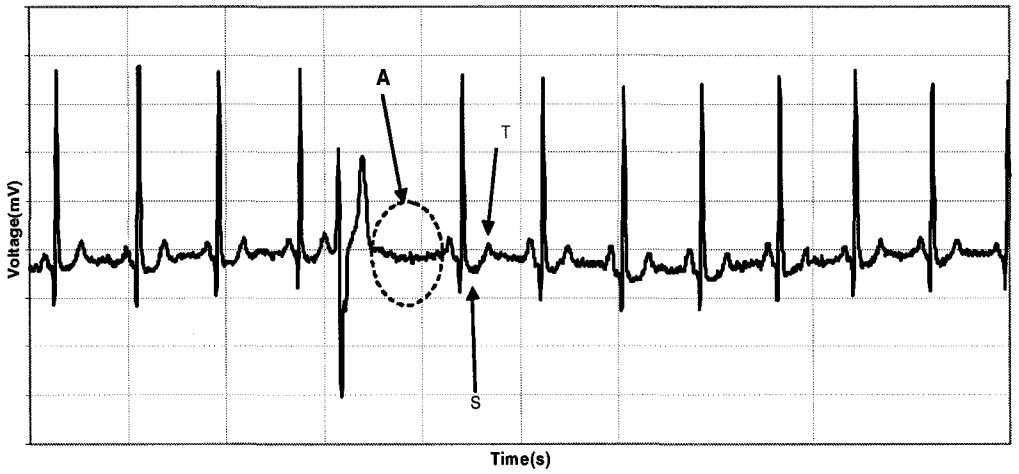
FAN has a different A section than the rest of the methods. The reason is that there are enough points in the slope before this section. Therefore the slope changes slowly and remains below the FAN threshold. AZTEC converts the A section to a plateau which is by its definition. CORTES is better than AZTEC, but it did not select the correct turning points. LA has a smooth line and correct S and T waves.

CCSP selects the turning point almost the same as the TP algorithm while its compression ratio is 10 times less than TP.

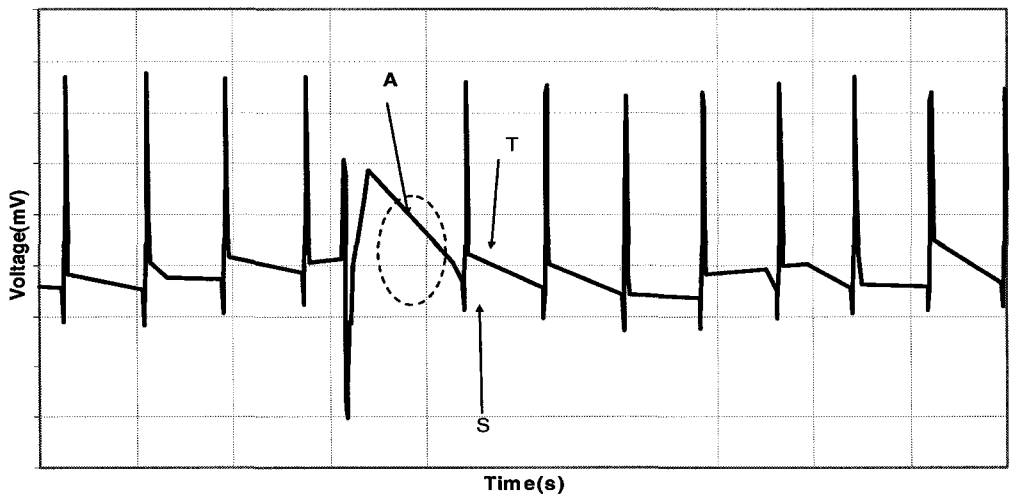
FAN, AZTEC and CORTES have chosen wrong S and T waves. However the rest of the methods have better selections, 5% does not seem to be a good compression ratio for any of the methods.

5.6.1.2 T6 - compression ratio of 10%

As it can be observed in Figures 5.17 to 5.20 with a compression ratio of 10% more similar graphs to the original one can be drawn. Total performance of all the methods have been improved. AZTEC and CORTES lack from the capability of choosing the correct turning points. Fan has a turning point in section A which is not part of the original graph. It happened because that point did not meet the threshold condition and therefore FAN selected it. A few points after that another point was out of the threshold and FAN picked that point. LA has the same pattern

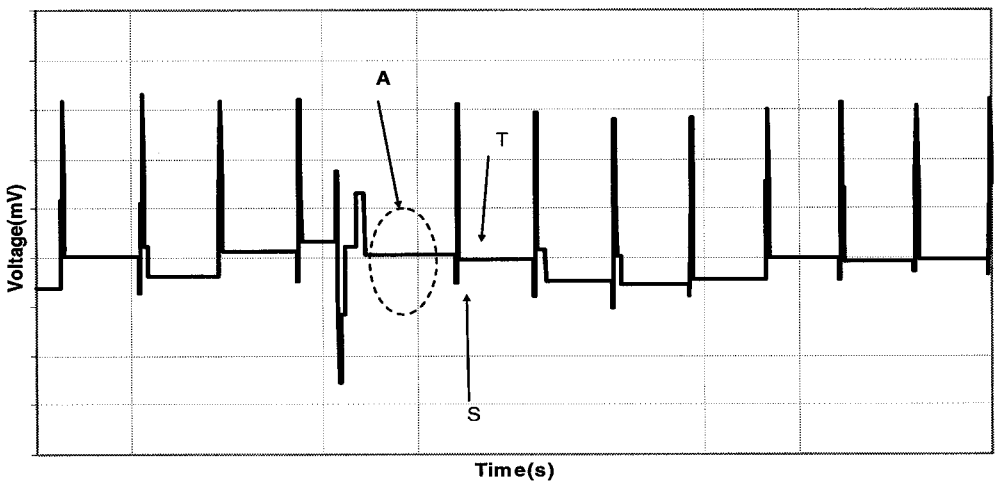


(a) Original Signal

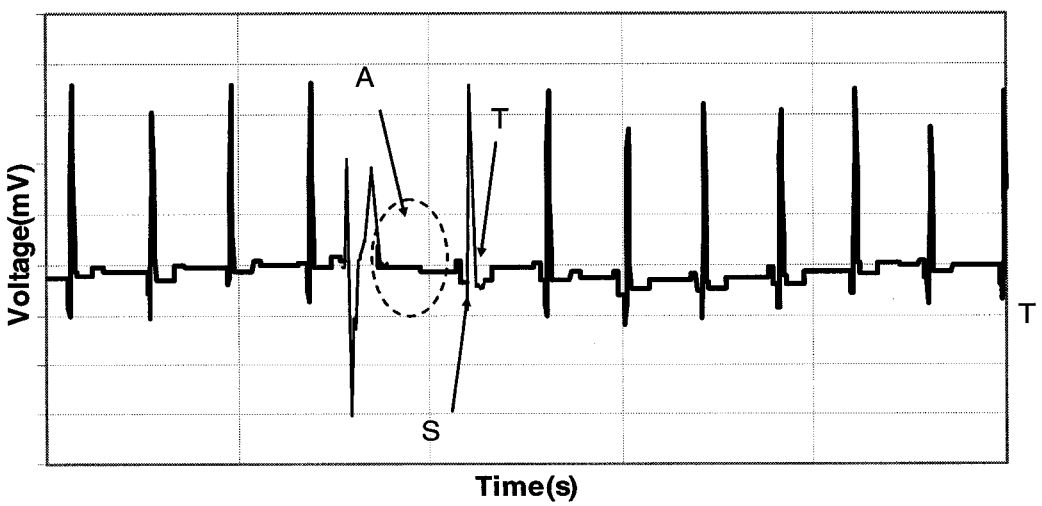


(b) FAN

Figure 5.13: Compression ratio of 5% for T6-part 1

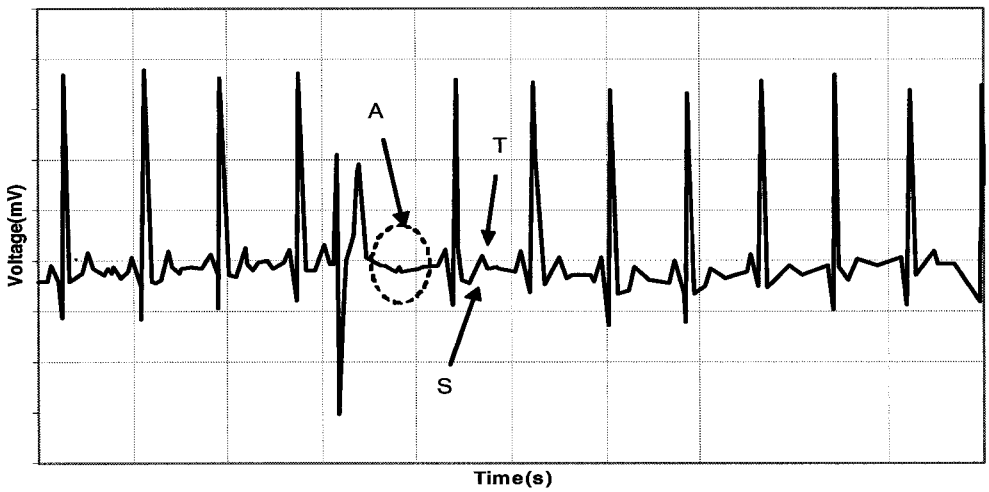


(a) AZTEC

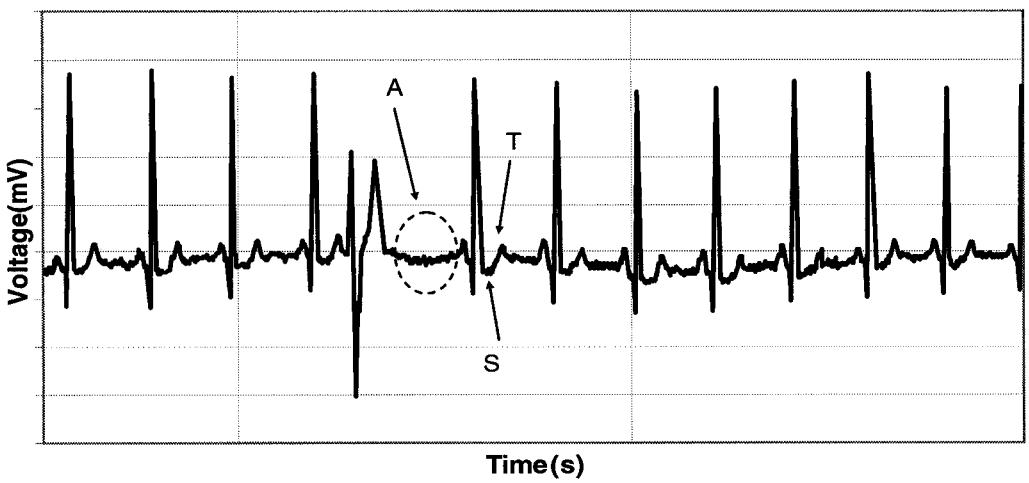


(b) CORTES

Figure 5.14: Compression Ratio of 5% for T6-part 2

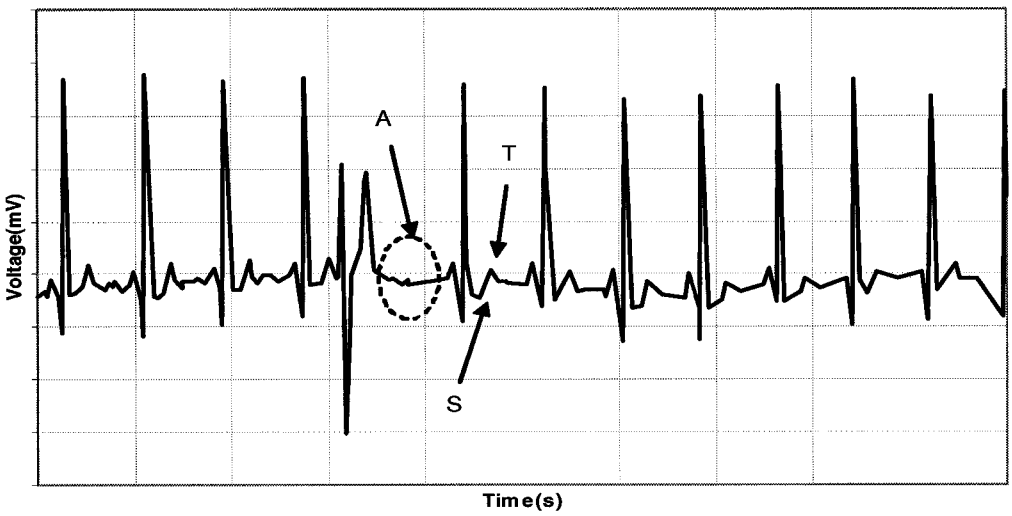


(a) CCSP

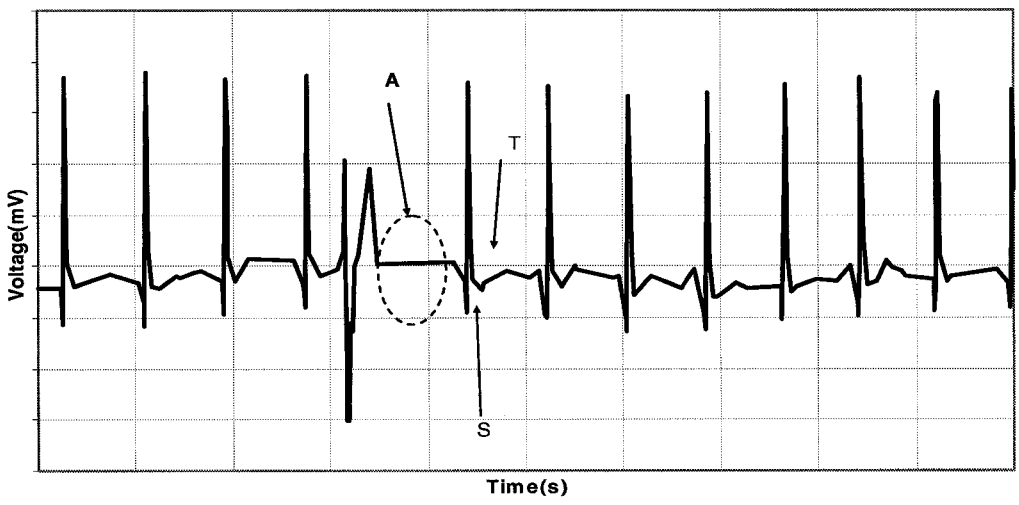


(b) TP with compression ratio of 50%

Figure 5.15: Compression Ratio of 5% for T6-part 3



(a) MCCSP



(b) LA

Figure 5.16: Compression Ratio of 5% for T6-part 4

in section A as the original graph has, but with less detail. The S and T wave are almost correct. CCSP and MCCSP are the most similar ones to the original graph. TP has the best graph, but the compression ratio is five times the rest.

5.6.2 Detailed Inspection

LA and MCCSP seem to work very well at compression ratio of 10%. In order to have a precise detailed inspection, a short segment of T1 has been chosen. The graph behavior of each method has been discussed followed by two example of signal analysis. PR interval and duration of QRS complex has been chosen since they have a specific mathematical value. The other criteria of signal analysis usually must be evaluated by a physician.

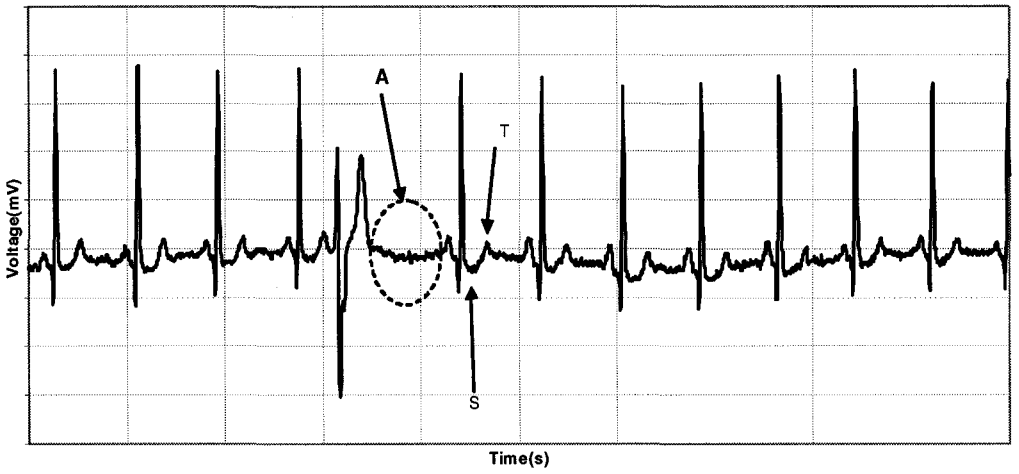
5.6.2.1 T1 - Compression Ratio of 10%

The reconstructed signal is shown at a compression ratio of 10% in Figure 5.21. The original signal is also included.

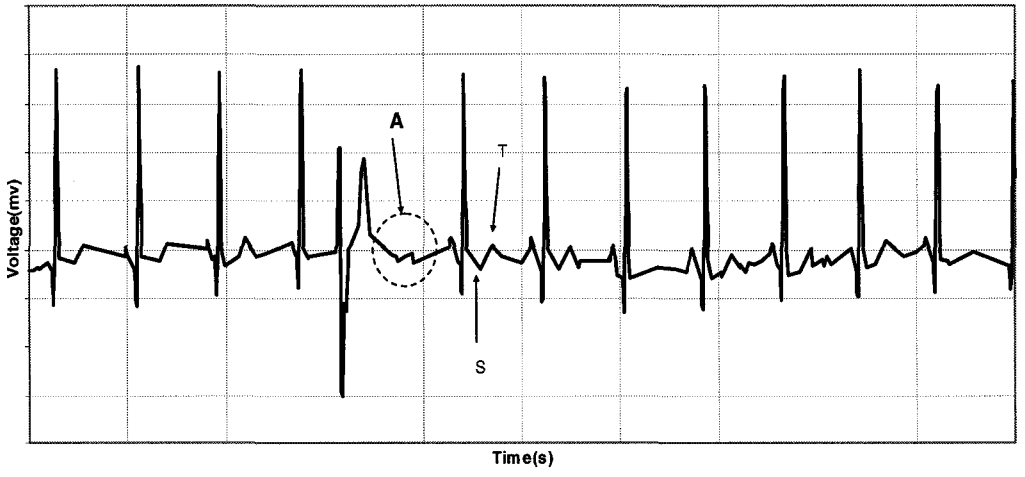
As can be seen in Figure 5.21, CCSP has a slightly lower R point than the original graph and AZTEC has a very low R point than other methods. FAN, AZTEC and CORTES have bad choices for the Q point. S has been chosen with almost the same voltage value, but different ST slopes. P and Q are the same points in FAN, AZTEC and CORTES.

As discussed previously, TP can achieve a compression ratio of 0.5 and therefore it has the most similar graph to the original one. MCCSP is the most similar graph and is selected all the marked points correctly. Although CCSP has chosen PQ and S acceptable, the R point is below the real one. The graph generated by LA seems to be accurate and to have all the waves. FAN, AZTEC and CORTES have chosen at leat one wrong point in the graph.

PR Interval Suppose the PR interval is necessary in a medical study. Table 5.6 shows the PR interval values from Figure 5.22 and the percentage of its difference with the real PR interval which has a value of 0.12.

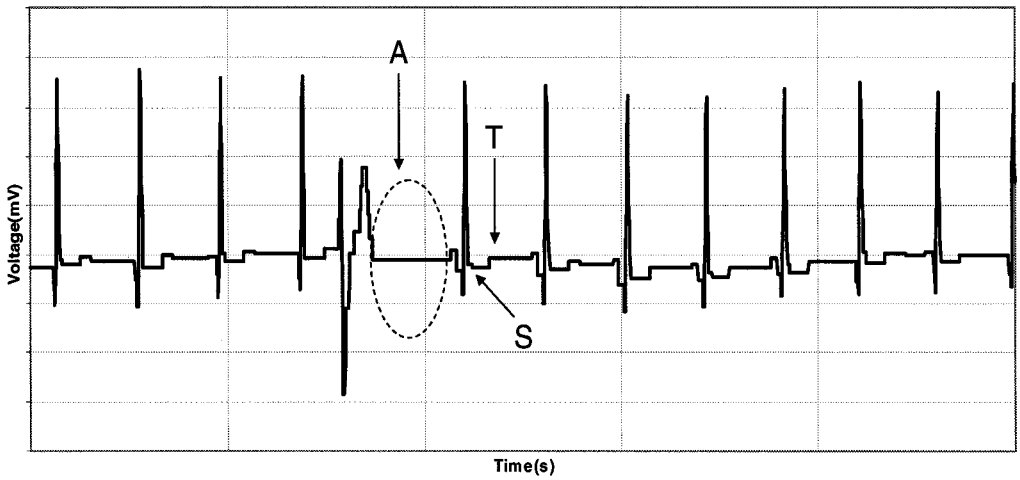


(a) Original Signal

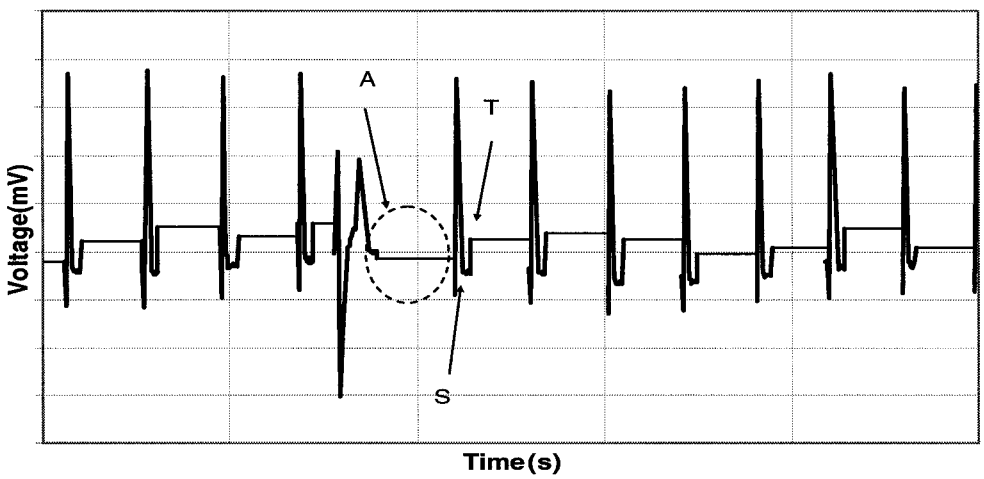


(b) FAN

Figure 5.17: Compression Ratio of 10% for T6- part 1

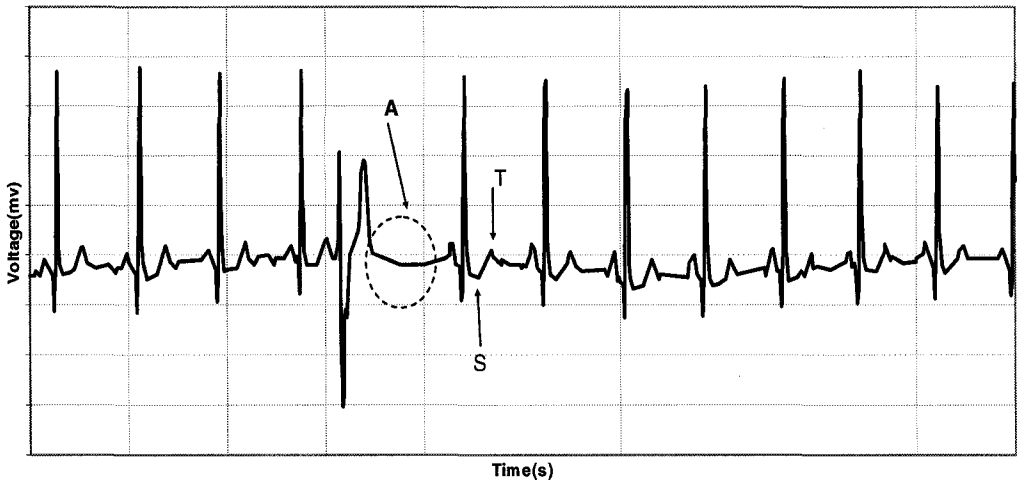


(a) AZTEC

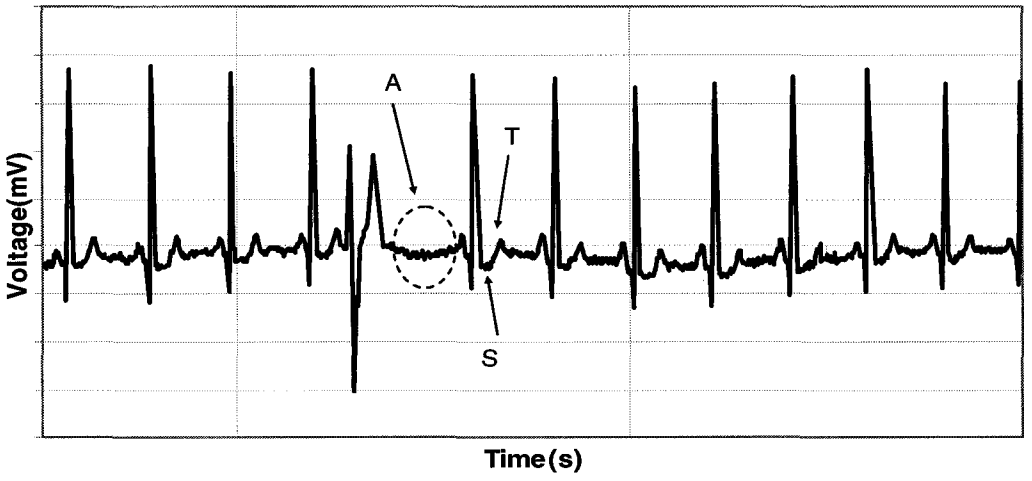


(b) CORTES

Figure 5.18: Compression Ratio of 10% for T6- part 2

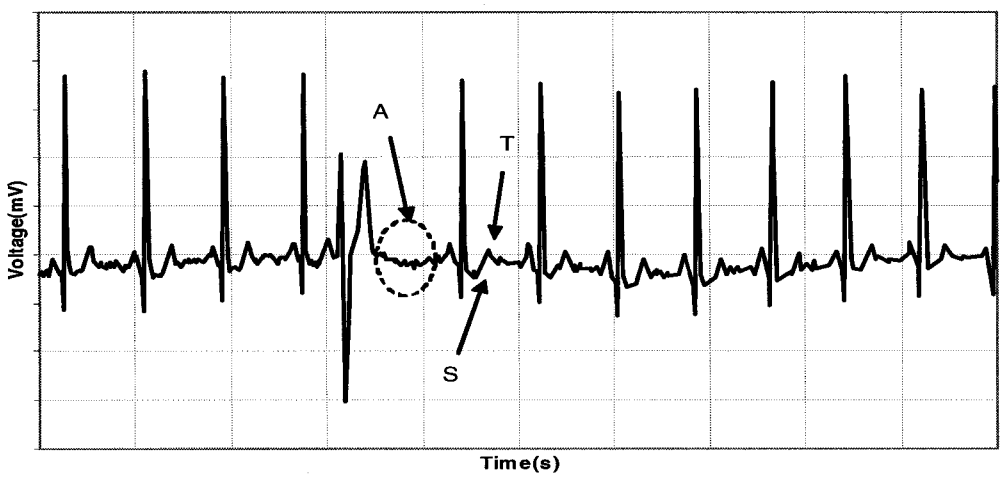


(a) CCSP



(b) TP with Compression ratio of 50%

Figure 5.19: Compression Ratio of 10% for T6- part 3



(a) MCCSP

(b) LA

Figure 5.20: Compression Ratio of 10% for T6- part 4

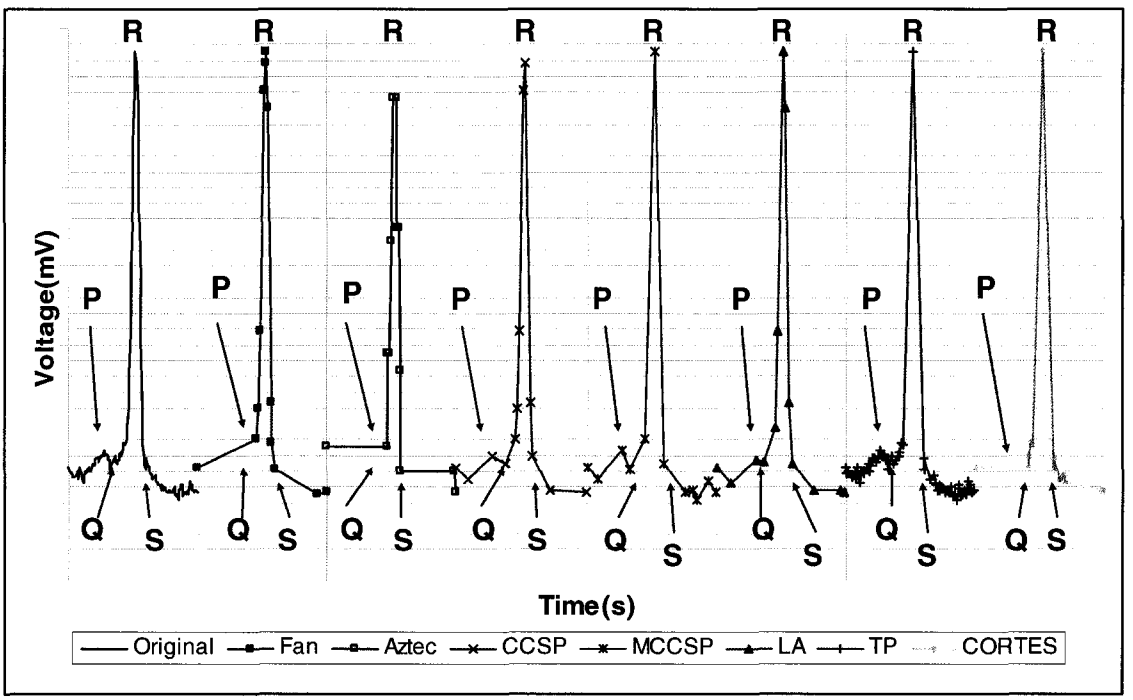


Figure 5.21: Compression Ratio of 10% for T1 (except for TP with CR=50%)

TP has the least value for PR interval, but the fact is that compression ratio for TP is five times the compression ratio for the rest of the methods. The PR interval value for MCCSP has 3.3% difference with the real value and the PR interval of LA is 6.7% different. CCSP has a larger value for PR interval. However CORTES has a better PR interval than FAN and AZTEC, all three of them have more than 50% difference with real value. Since reason of them have selected the wrong turning points.

Duration of QRS Complex

The duration of the QRS complex is a constant value of 0.12 seconds according to Section 2.2.

Table 5.7 shows that none of the methods can achieve the exact amount of QRS

Method	PR Interval Duration(s)	Relative PR Interval
Real	0.12	100%
TP	0.124	103.3%
MCCSP	0.124	103.33%
LA	0.128	106.67%
CCSP	0.148	123.33%
CORTES	0.204	170.0%
FAN	0.224	186.66%
AZTEC	0.232	193.33%

Table 5.6: PR Interval from Figure 5.21

which is not a surprise with this low compression ratio of 10%. LA is 0.008 seconds short and MCCSP is 0.008 seconds long which are the closest values to the real duration of QRS complex. Considering the fact that each small square (every 1mm) in ECG graph is 0.04 seconds, 0.008 seconds is shown 0.02mm which is hard to capture by eye. It can be concluded that LA and MCCSP can achieve the correct duration of QRS complex. This value for TP is 0.092.

Method	Original	FAN	AZTEC	CCSP	MCCSP	LA	CORTES
QRS duration(s)	0.12	0.076	0.056	0.104	0.128	0.112	0.096

Table 5.7: QRS Duration from Figure 5.21

5.6.2.2 T1 - Compression Ratio of 17%

The same graph with a compression ratio of 17% is shown in Figure 5.23

The reconstructed graph has been improved in CCSP, MCCSP and LA. CORTES is slightly different. FAN and AZTEC have selected the same P and Q the same point as they did for the smaller compression ratio. They both have chosen extra points where it seemed unnecessary. The R point is in the correct location for all the methods.

PR Interval The original PR interval value is 0.12 in the original graph. The PR interval value for CCSP, MCCSP is the same as LA with only 6.7% difference as shown in Figure 5.8 and Table 5.24. CORTES has a better PR interval than FAN

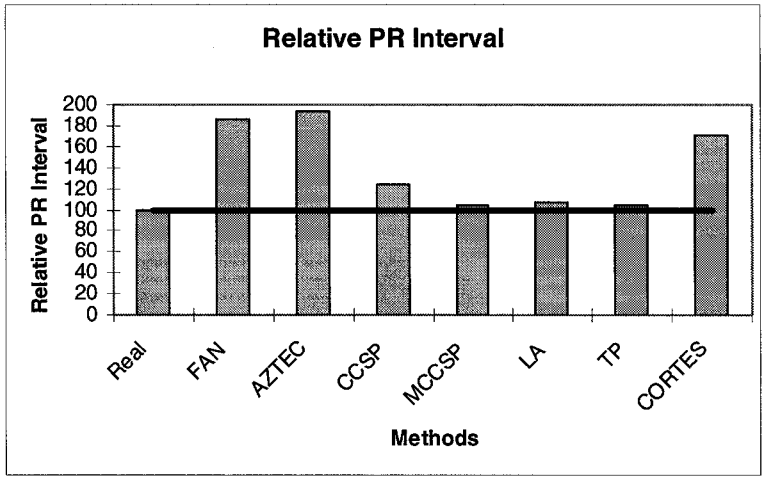


Figure 5.22: Demonstration of table 5.6

and AZTEC. Since CORTES has chosen at least one of the turning points in the high frequency area correctly.

Method	PR Interval Duration(s)	Relative PR Interval
Original	0.12	100%
TP	0.124	103.3%
CCSP	0.128	106.7%
MCCSP	0.128	106.7%
LA	0.128	106.7%
CORTES	0.151	125.83%
FAN	0.168	140.0%
AZTEC	0.212	176.66%

Table 5.8: PR Interval from Figure 5.23

Duration of QRS Complex

Table 5.9 shows the none of the methods can achieve the exact amount of QRS, however, LA is 0.004 seconds and CCSP is 0.012 seconds shorter and MCCSP is 0.016 seconds longer than the original value of 0.12 seconds. As explained in Section 5.6.2.1, every 1mm in ECG graph is 0.04 seconds, 0.004 seconds is shown 0.1mm and 0.016

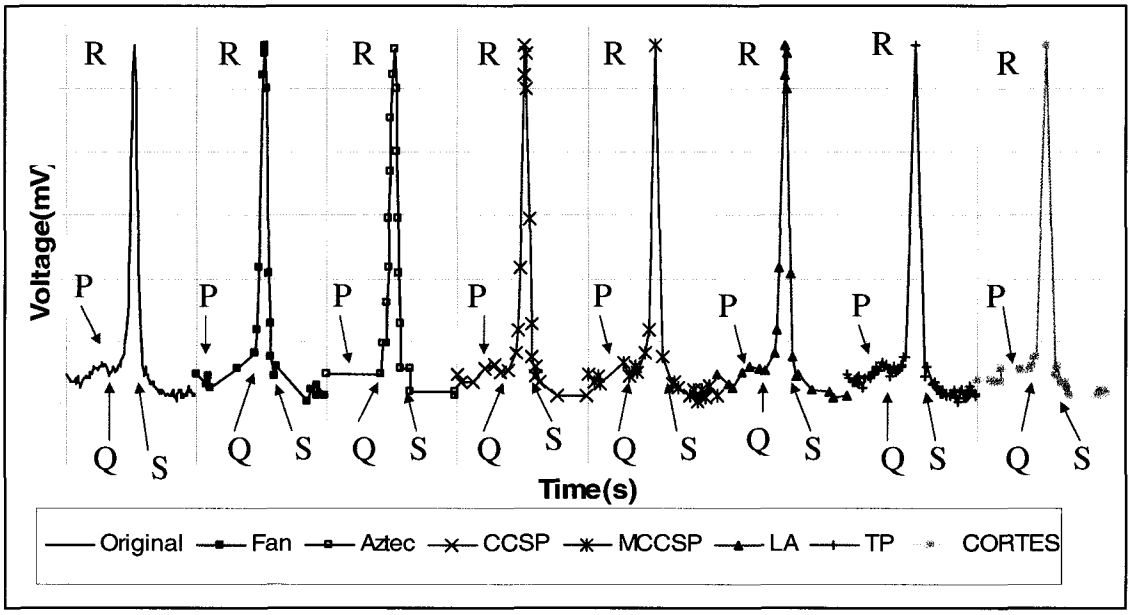


Figure 5.23: Compression Ratio of 17% for T1

seconds is shown 0.4mm which are not easy to see. Therefore LA and MCCSP have calculated the duration of QRS complex correctly. This value for TP is 0.092.

Method	Original	FAN	AZTEC	CCSP	MCCSP	LA	CORTES
QRS duration(s)	0.12	0.076	0.076	0.108	0.136	0.116	0.092

Table 5.9: QRS Duration from Figure 5.23

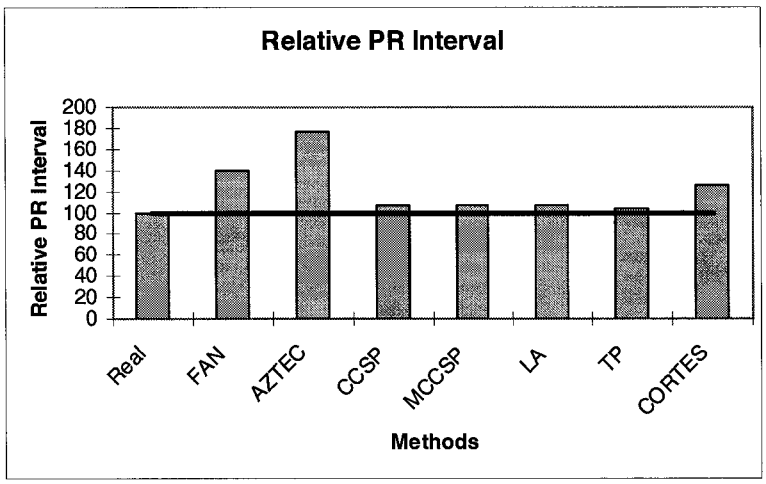


Figure 5.24: Demonstration of table 5.8

5.7 Evaluation Based On Execution Time

An important aspect of a compression algorithm is the ability to run in real time. In order to evaluate the CCSP algorithm, some benchmarks are presented here, i.e., some execution times for the algorithm for different compression ratios run on a specific machine.

Suppose N represents the total number of points in original graph and M is the total number of points in reconstructed graph. Assume there are Q points between two turning points of the original signal. The complexity of CCSP is $O(MN^2)$, for FAN is of $O(N)$, for AZTEC O(N), for MCCSP $O(N + MQ^2)$ and for LA is $O(N)$.

The execution time tests are based on the average of running tests T2, T4, T6 and T10[2] for 100 times.

Figure 5.25 shows the execution time is very small for FAN, LA and AZTEC methods and almost independent of the compression ratio.

[2]Only 1000 sample points are used when comparing CCSP and MCCSP. It takes 145 seconds for CCSP to have a compression ratio of 0.6 for 4 seconds original data

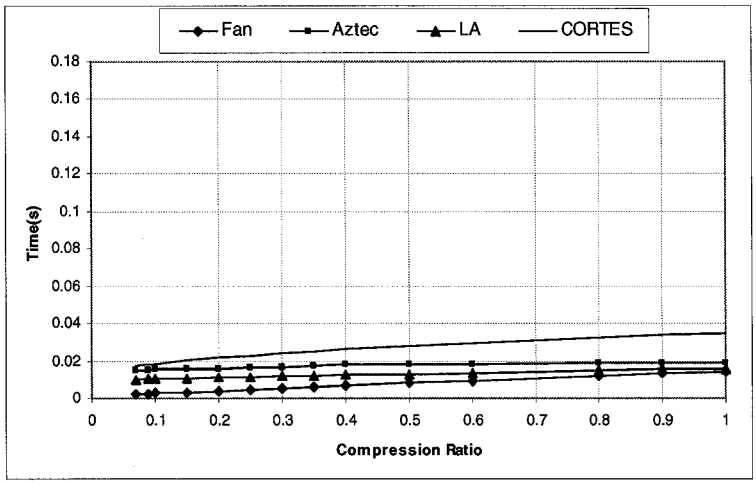
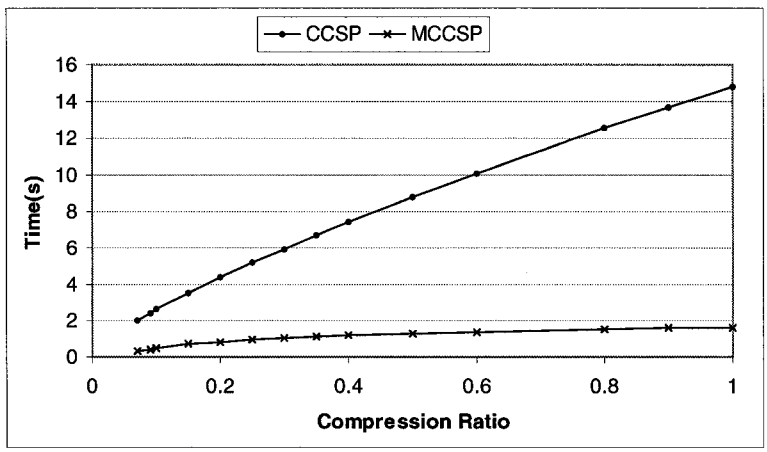


Figure 5.25: Execution time - part 1

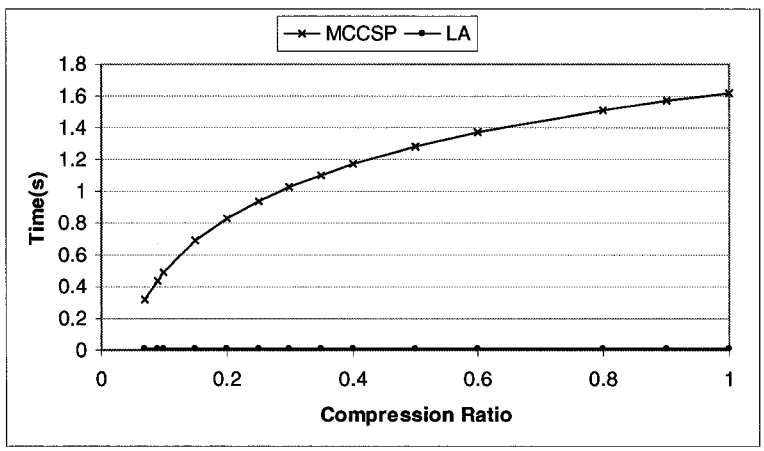
On the other hand, considering Figure 5.26 the execution time for MCCSP has been improved in compare to the original CCSP algorithm. The expected time of MCCSP is larger than LA as it can be seen in Figure 5.26(b), but for a compression ratio of 30%, it takes 1 second for MCCSP to compress the data which is very small in compare to original data duration of 60 seconds. Both MCCSP and LA can be used in on-line applications.

One of the measurements of the quality of the compressed signal is PRD. Figure 5.27 shows that almost all the algorithms with the exception of CCSP are independent of PRD. The reason is when the algorithm process a point, regardless of selection decision it processes the next point and the complexity of these algorithms are linear and completely independent of PRD. In the other word PRD does not affect the selection decision.

Figure 5.27(b) compares CCSP and MCCSP. The main idea in CCSP is minimizing the PRD. Therefore to achieve a small PRD, more points must be processed several times. Since MCCSP minimizes the area instead of PRD, MCCSP is linear in Figure 5.27(b) but CCSP is not.

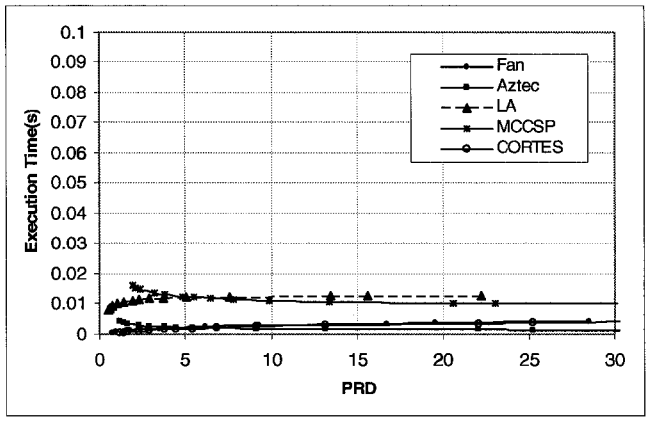


(a) MCCSP and CCSP average Time(input size=1000 points)

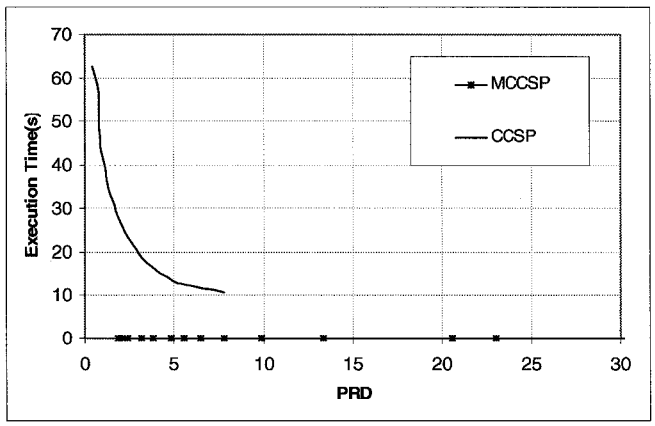


(b) MCCSP and LA average Time

Figure 5.26: Execution time- part 2



(a) Average Time



(b) MCCSP and CCSP average Time(input size=1000 points)

Figure 5.27: Execution time versus PRD

5.8 Summary

In this chapter, the results of seven ECG signal compression techniques have been discussed. Each algorithm has been tested with the MIT-BIH Arrhythmia database. All of the reconstructed graphs have been evaluated not only in terms of PRD and maximum error, but also they are visually compared to the original graph. The execution time of each compression technique has also been studied.

MCCSP is faster than CCSP but it does not have good results in terms of PRD and maximum error. Its execution time is considerably improved and it preserves the quality of the signal in the same way as CCSP.

The strength of the LA algorithm has been shown in all four evaluation categories. It has the lowest maximum error and PRD among all other methods with the exception of CCSP. LA is faster than CCSP and MCCSP and in close proximity to AZTEC, FAN, CORTES and TP. The LA method also generates very similar graphs to the original one which can be used in signal analysis even in small compression ratios.

Chapter 6

Summary and Conclusion

In this thesis, we have investigated a simple mathematical approach to signal compression. The focus has been on designing a simple, accurate and fast methods to be used in on-line applications.

The error is being minimized based on the area between the original and reconstructed graphs with respect to the given constraints, not in terms of minimizing the difference value between each reconstructed and original point.

By the very nature of this approach, the problem is solved in a more optimal manner in comparison to other traditional methods. Moreover, this method is not computationally expensive and it does not involve any preprocessing.

We have also modified a computationally expensive algorithm which provided high quality compression. The error definition is different than the original algorithms and by using some simple techniques the running time of the algorithm has been improved.

6.1 Contributions

1. A new technique for signal compression - the Least Area method (LA) - was developed in Chapter 3. The methodology is based on processing the signal

as it is being received by the device, thereby making it suitable for on-line applications. We also showed that the compression problem can be solved in a computationally effective manner, with the expected linear complexity. This algorithm receives area threshold as input and its output is a compression ratio. Defining the critical turning points in the algorithm prevents it from having a quadratic complexity. Instead of limiting the number of points in the solution, we rather develop an approach finding the critical turning points and limiting the area between each two turning points.

2. Some modifications for one of the best compression methods, CCSP, has been proposed in Chapter 4. These modifications were meant to reduce the execution time and therefore a new definition of error is presented and the signal is processed with a limited number of points. Although CCSP needs a huge memory space to store all the original data, MCCSP does not occupy a lot of memory space; it deletes the data from memory as soon as it processes the block. It is also much faster and has good quality results. The input of this algorithm is compression ratio.

3. In Chapter 5, a program was developed for testing the MCCSP and LA methods and their results are compared to five well known algorithms; FAN, AZTEC, CORTES, TP and CCSP. CCSP has the minimum possible error, but its complexity is cubic. Fan and Aztec are linear algorithms, but their result is poor when the compression ratio is very low. Performance of CORTES is between FAN and AZTEC. The nature of TP algorithm prevents the compression ratio from being below a certain amount.

6.2 Conclusions

Several conclusions may be drawn from the experiments. Below is a list of the main conclusions:

- Looking at the compression problem from a traditional view is still useful in

modern applications. In Chapter 3 we introduce the Least Area method and in Chapter 5 we discussed the developed LA algorithm and found that it generally performs much better than both traditional and more recently developed time domain ECG compression methods.

- In terms of PRD, the LA results are more accurate than FAN, AZTEC, CORTES and MCCSP for compression ratios of less than 30%, but it is less accurate than CCSP. For compression ratios of more than 30%, all the methods generate similar results.
- The maximum error of the LA method for compression ratios of less than 20% is almost 40% more than that of CCSP but less than other methods.
- Visual inspection indicates the LA algorithm generates a smooth graph, but still chooses the critical points correctly. CCSP and MCCSP generate the closest graph to the original one, but AZTEC and CORTES make a step like graph (specially in low compression ratios) and FAN does not choose the critical points correctly.
- When it comes to the running time comparison; CCSP (the best algorithm in terms of PRD, maximum error and visual inspection) takes a long time to process even a small graph. LA runs in linear time and it has almost no difference with AZTEC, FAN, CORTES and TP in complexity.

• CCSP method is generating the most similar graph to the original one in compare with the other time domain methods. However, more modifications are required in order to achieve better complexities. In Chapter 4 some modifications are forced on this algorithm. These constraints have slightly improved the execution time and lead us to a new perspective for more traditional methods. The coding experiments have shown that the MCCSP method can be used in some on-line applications with some strategies and has its own strength.

- In terms of PRD the MCCSP results are the worst results. By increasing the compression ratio to 30% its PRD is lowered, but larger than the

PRD of CCSP. For compression ratio more than 10%, the PRD is very large. High frequency parts of the signal are the reason for this high PRD. MCCSP selects as few points as possible and this number is usually very small in smaller compression ratios. However the reconstructed signal is very similar to the original one, the difference between the reconstructed voltage value and the original is large and it makes the PRD rise as explained in Figure 5.3.

- The maximum error of the MCCSP method is more than all the methods and it happens in high frequency sections.
- Visual inspection shows MCCSP is the closest to the original graph in that it selects the critical points and has enough detail even in small compression ratios. The signal analysis showed that MCCSP is even more accurate than CCSP in lower compression ratios.
- Running time has been improved in MCCSP (over the CCSP); but it is still slower than the other methods.

- Considering all of the comparison areas, LA generates the best results for an on-line algorithm with very good accuracy.

- We have investigated the use of polynomial of order one in compression of ECG signals. From this we draw the conclusion that applying a polynomial of order one, i.e., linear line segments, in approximating the signal gives acceptable results. The main purpose of linear interpolation is to keep the algorithm simple and fast. Increasing the order of the polynomial from one to two, may increase the performance of the compression scheme, but it will definitely affect the running time and it indicates that this may not be worthwhile.

6.3 Directions for Future Research

Approximating signals by the use of polynomials of varying orders

In this thesis we have compressed signals by linear interpolation. However it is useful to keep the execution time very low, but some pieces of the signal may be represented most efficiently by a second order polynomial, while others, where there is less activity, may be better approximated using linear line segments. Coding with a varying order of the polynomial involves one extra parameter, namely the order of the polynomial. Most likely, this parameter will have a small dynamic range and will thus hopefully not give a big contribution to the total compression ratio. It would be fruitful to examine an approach based on this idea further.

Extraction of a Variable Number of Samples from Each Signal Block

As we have seen in the experimental section, the MCCSP algorithm faces a challenge in coping with the maximum error. This may be partly due to the fact that it is only allowed to extract the same predefined number of samples from each block of the signal. If the signal contains high frequencies, the peaks resulting in the overall lowest cost may be left out. One such peak in a long signal sequence, results in a high maximum error for the total reconstructed signal. One way to account for this effect could be by allowing the MCCSP algorithm to extract a variable number of samples from each block of the signal. This could be implemented in the version of the MCCSP algorithm with the maximum error bound incorporated.

The same technique can be used in LA algorithms as well. In the algorithm presented in this thesis, a second variable threshold was used. Variable number of samples can replace the proposed threshold in lower frequency sections also and can effectively increase the details in the constructed graph.

Appendix A

Test Databases

The MIT-BIH Arrhythmia database [19] was used to evaluate the algorithms. These test are selected based on different diseases, dissimilarities to each other and pattern of changes.

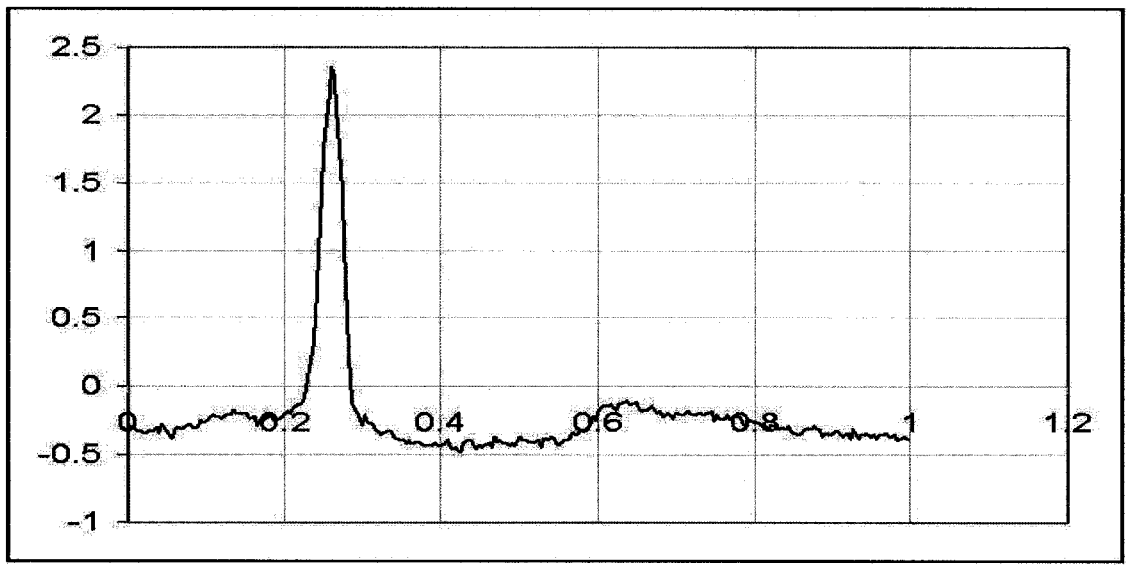


Figure A.1: MIT-BIH CV ventricular database

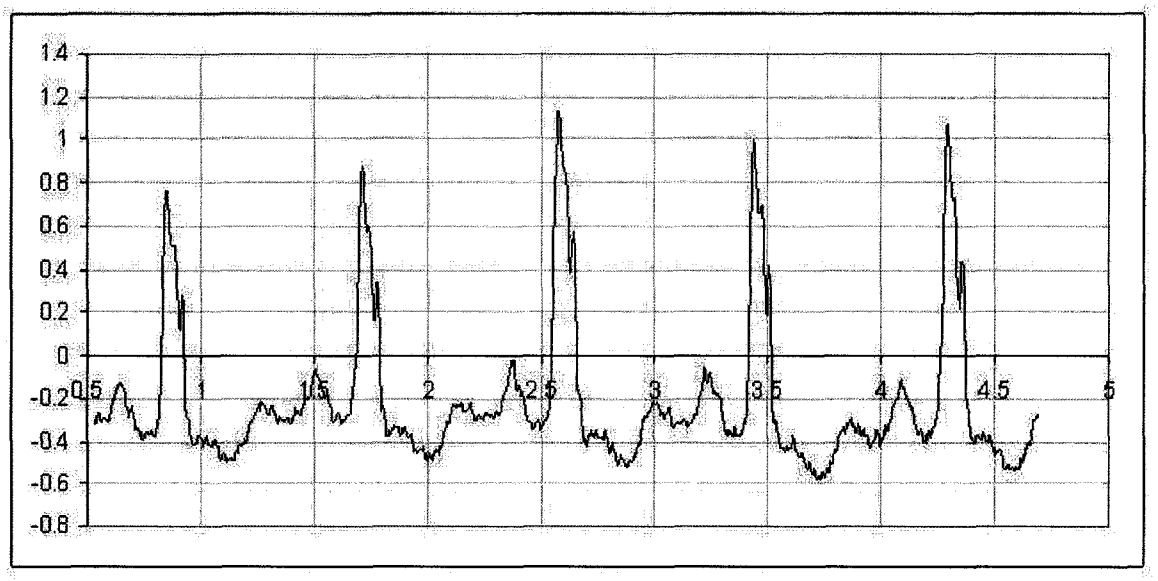


Figure A.2: MIT-BIH BIDMC database

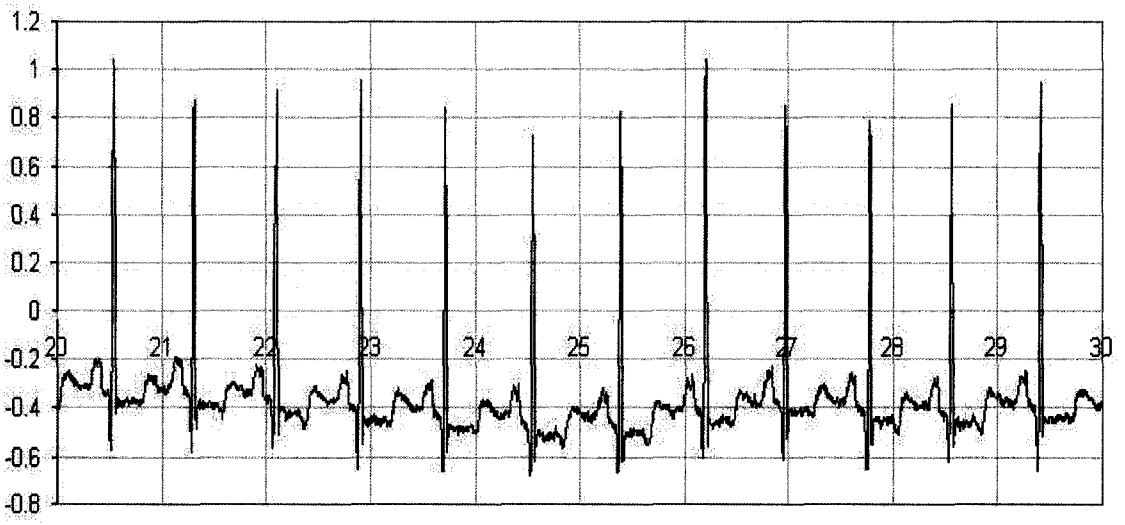


Figure A.3: MIT-BIH Arrhythmia database

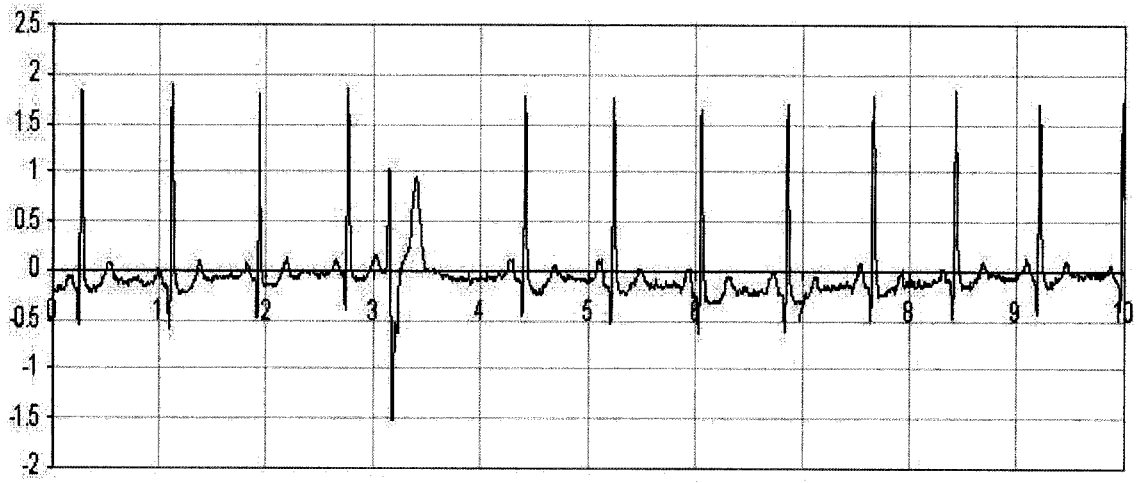


Figure A.4: MIT-BIH AHA database

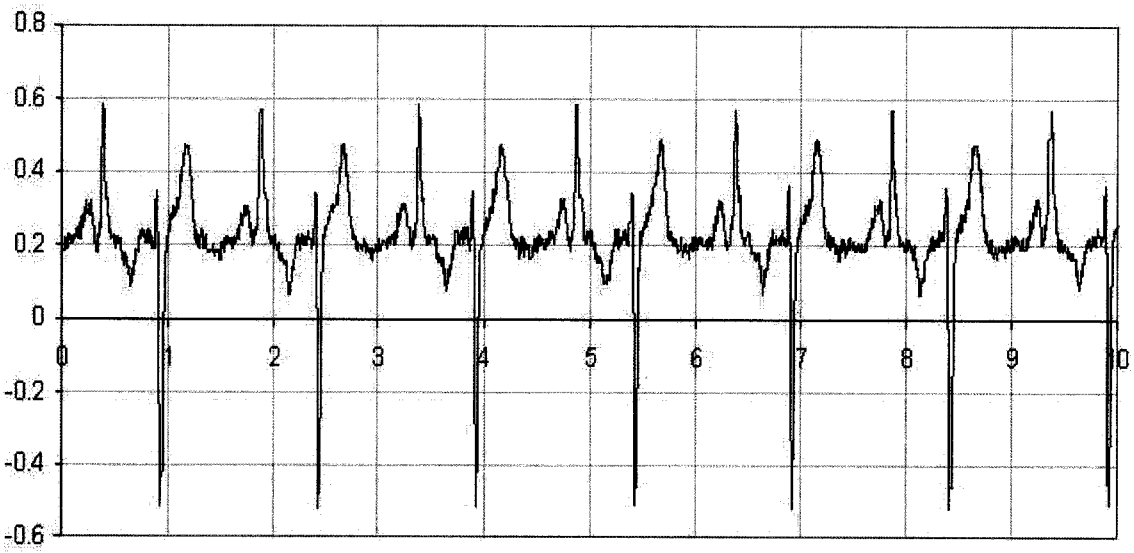


Figure A.5: MIT-BIH ANSI AAMI database

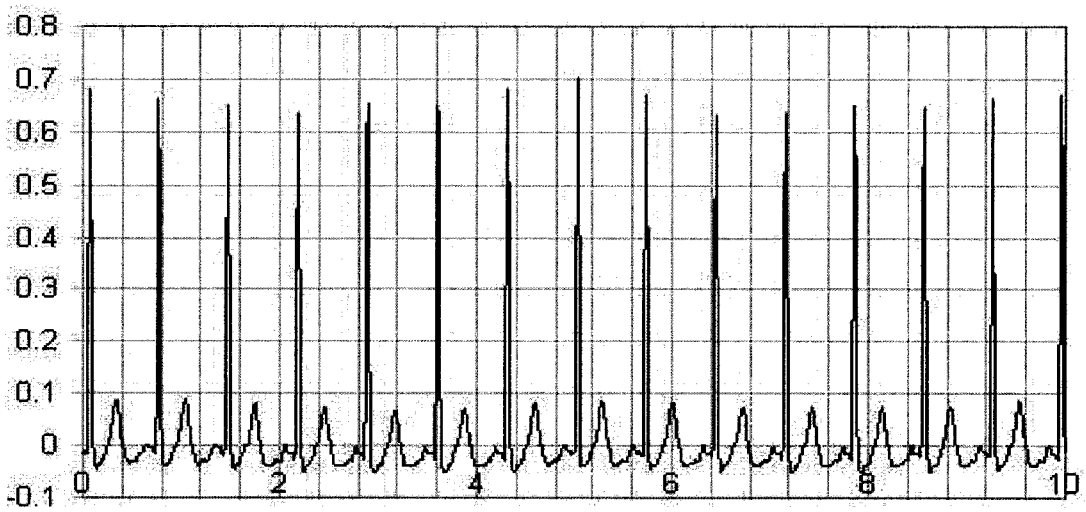


Figure A.6: MIT-BIH PAF Prediction database

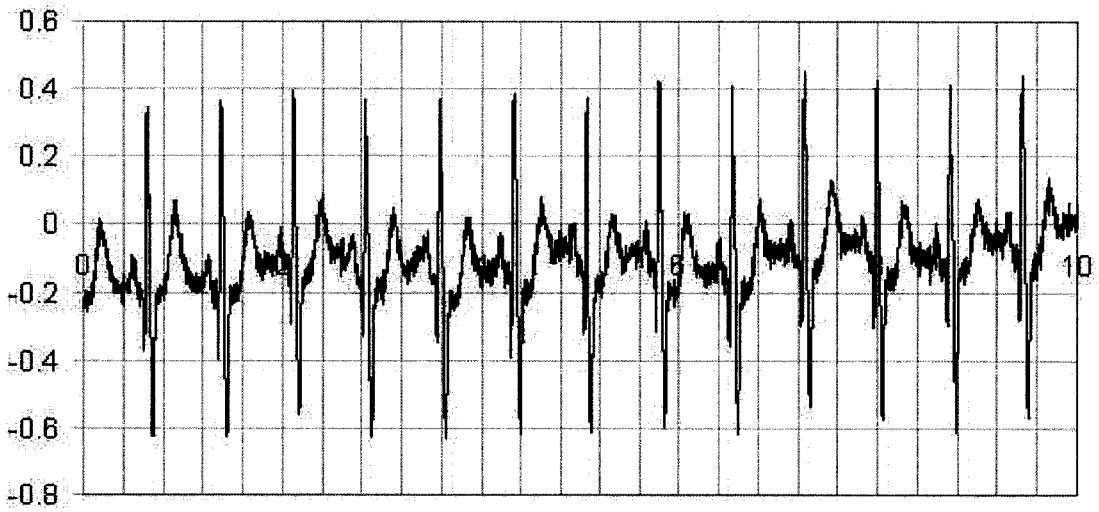


Figure A.7: MIT-BIH PRB Diagnostic database

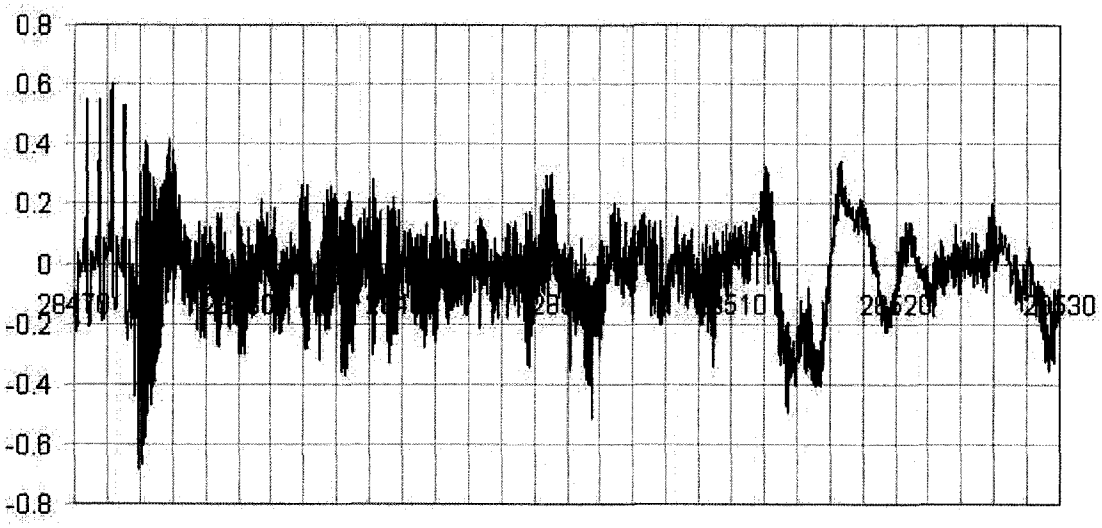


Figure A.8: MIT-BIH Sudden Cardiac Death Holter database

Bibliography

[1] S. O. Aase. Filter bank design for subband compression of ECG signals. In *Proc. Norwegian Signal Processing Symp*, pages 113–118, 1995.

[2] J. Abenstein and W.Tompkins. New datareduction algorithm for realtime ecg analysis. *IEEE Trans. on Biomedical Engineering*, BME-29:43–48, 1982.

[3] J.R. Cox., F.M.Nolle, H.A. Fozzard, and G.C. Oliver Jr. Aztec – a preprocessing program for real-time ecg rhythm analysis. *IEEE Trans. on Biomedical Eng.*, 15:128–129, 1968.

[4] D.A. Dipersio and R.C.Barr. Evaluation of the fan method of adaptive sampling on human electrocardiograms. *Medical and Biological Engineering and Computing*, pages 401–410, September 1985.

[5] A. Djohan, T. Q. Nguyen, and W. J. Tompkins. Ecg compression using discrete symmetric wavelet transform. *in Proc. of 17th Int. IEEE Med. Biol. Conf.*, 1:167 –168, 1997.

[6] D.H. Douglas and T.K. Peucker. Algorithms for the reduction of the number of points required to represent a line and its caricature. *The Canadian Cartographer*, 10(2):112–122, 1973.

[7] K. Duda, P. Turcza, and T.P. Zielinski. Lossless ecg compression with lifting wavelet transform. *Proceedings of the 18th IEEE*, 1:640–644, May 2001.

[8] L.W. Gardenhire. *Data Compression for Biomedical Telemetry*, chapter 11. C.A.Caceres, New York, academic edition, 1965.

[9] D. Haugland, J.G.Heber, and J.H.Husy. An optimum time-domain ecg data compression scheme. *in Proc. BIOSIGNAL-96*, pages 177–179, June 1996.

[10] J.G. Heber, D. Haugland, and J.H. Husey. An efficient implementation of an optimal time domain ecg coder. *Proceedings of the 18th Annual International Conference of the IEEE*, 4:1384 – 1385, Oct 1996.

[11] R. N. Horspool and W. J. Windels. An lz approach to ecg compression. *Proc. 7th IEEE Symp.Comp.-Based Med. Sys*, pages 71–76, June 1994.

[12] R.J. Huszar. *Basic Dysrhythmias: Interpretation Management.* Mosby Lifeline, St. Louis, Missouri, 2nd edition, 1994.

[13] S. Jalaleddine, C. Hutchens, R. Strattan, and W. Coberly. Ecg data compression techniques - a unified approach. *IEEE Trans. on Biomedical Engineering*, 37:329–343, 1990.

[14] D. Jenkins and S.J. Gerred. *ECGs by Example.* Churchill Livingstone, 1997.

[15] R. Klabunde. *Cardiovascular Physiology Concepts.* Lippincott Williams Wilkins, Baltimore, 2004.

[16] W. S. Kuklinski. Fast walsh transform data-compression algorithm; ecg applications. *Medical 4 Biological Engineering 4 Computing*, 21:465–472, July 1983.

[17] C. Lamberti, M. Zagnoni, R. Degani, and G. Bortolan. Evaluation of algorithms for real-time ecg data compression. *Computers in Cardiology 1990. Proceedings*, pages 399 – 402, September 1990.

[18] J. Malmivuo and R. Plonsey. *Bioelectromagnetism - Principles and Applications of Bioelectric and Biomagnetic Fields.* Oxford University Press, New York, 1995.

[19] G.B. Moody. Mit-bih arrhythmia database. Internet database, Harvard MIT Division of Health Sciences and Technology, 2005.

[20] G.B. Moody., R.G. Mark, and A.L. Goldberger. Evaluation of the trim ecg data compressor. *in Proc. Computers in Cardiology*, pages 167–170, 1989.

[21] G.B. Moody., K. Soroushian, and R.G. Mark. Ecg data compression for tapeless ambulatory monitors. *Computers in Cardiology*, 14:467–470, 1987.

[22] W. Mueller. Arrhythmia detection program for an ambulatory ecg monitor. *Biomed. Sci. Instrument*, 14:81–85, 1978.

[23] R. Nygaard and D.Haugland. Complete coding scheme using optimal time domain ecg compression methods. *in Proc. Of Eur. Signal Proc Conf. (EUSIPCO)*, pages 2473–2476, September 1998.

[24] S. Olmos, M. Milln, J. Garcfa, and P. Laguna. Ecg data compression with the karhunen-love transform. *in Computers in Cardiology Proceedings*, pages 253–256, September 1996.

[25] B.R.S. Reddy and I. S. N. Murthy. Ecg data compression using fourier descriptions. *IEEE Yans. on Biomedical Engineering*, 33:428–434, 1986.

[26] A. M. Safi. Use of intracoronary electrocardiography for detecting "st-t", "qtc", and "u" wave changes during coronary balloon angioplasty. *Heart Disease: A Journal of Cardiovascular Medicine*, 5:73–76, 2003.

[27] T. Schimming, M. Ogorzalek, and H. Dedieu. A nonlinear dynamical model for compression and detection of ecg data. *In Proceedings EUSIPCO-98*, 1998.

[28] E.V. Stansfield, D. Harmer, and M.F. Kerrigan. Speech processing techniques for hf radio security. *Communications, Speech and Vision, IEE Proceedings I*, 136(1):25 – 46, Feb. 1989.

[29] S.C. Tai. Slope - a real-time ecg data compressor. *Medical Biological Engineering Computing*, 29:175–179, March 1991.

[30] S.C. Tai. Aztdis - a two phase real-time ecg data compressor. *Journal of Biomedical Engineering*, 15:510–515, Nov. 1993.

[31] Chr. Zywietz, G. Joseph, and R. Fischer. Compression of diagnostic resting electrocardiograms. In *Computers in Cardiology Proceedings*, pages 391–394, September 1990.