3 Biomedical signal processing

Chapter 3
Biomedical signal processing

Due to growing complexity of the biomedical examinations, comprehensive documentation and need for automation to reduce costs acquisition and processing of biomedical signals has become more and more important for the physician [1]. The high quality measurement techniques and mathematical tools are responsible for the progress in bio medical signal processing.

Model-based signal processing techniques based on a biophysical model of the underlying physiological process have been developed. Traditional signal processing techniques, like time-frequency domain or wavelet analysis are applied to bio molecular data for classification and pattern recognition.

The important processes are Filtering, averaging, compression and detection. The chapter gives a brief summary of traditional techniques for each of these processes.

3.1 Filtering:

Numbers of researchers were working independently on different applications of adaptive filters in late 1950s. The **least-mean-square (LMS) algorithm** emerged as a simple algorithm for the operation of adaptive transversal filters. Widrow and Hoff devised the LMS algorithm in 1959 in their study of a pattern recognition scheme known as the adaptive linear threshold logic element (Widrow and Hoff, 1960 Widrow, 1970). The LMS algorithm is a **stochastic gradient algorithm.**

It iterates each tap weight in the transversal filter in the direction of the gradient of the squared amplitude of an error signal with respect to that tap weight. As such, the LMS algorithm is closely related to the concept of stochastic approximation developed by Robbins and Monroe in statistics for solving certain sequential parameter estimation problems (Robbins and Monroe, 1951). The primary difference between them is that the LMS algorithm uses a fixed step-size parameter to control the correction applied to the tap weight from one iteration to the next, whereas in stochastic approximation method the step size parameter is made inversely proportional to time n or to a power of n. Another algorithm, closely related to the LMS algorithm, is the gradient adaptive lattice (GAL) algorithm (Griffiths, 1977, 1978); the difference between them is structural in that the GAL algorithm is lattice-based, whereas the LMS algorithm uses a transversal filter.

Godard made another major contribution to the development of adaptive filtering algorithms in 1974. He used **Kalman filter theory** to propose a new class a transversal filter to their optimum settings (Godard, 1974). Although, prior to this date, several investigators had applied Kalman filter theory to solve the adaptive filtering problem, Godard's approach is widely accepted as the most successful. This algorithm is referred to in the literature as the Kalman algorithm or Godard algorithm.

3.1.1 Filtering: General issues

The design of a **Wiener filter** requires a prior information about the statistics of the data to be processed. The filter is optimum only when the statistical characteristics of the input data match a prior information on which the design of the filter is based. A more efficient method is to use an **adaptive filter**. It is self designing in that the adaptive filter relies for its operation on a recursive algorithm, which makes it possible for the filter to perform satisfactorily in an environment where complete knowledge of the relevant signal characteristics is not available for example, in case of biomedical signals. The algorithm starts from some predetermined set of initial conditions, representing complete ignorance about the environment. Yet, in a stationary environment, one finds that after successive iterations of the algorithm, it converges to the **optimum Wiener solution** in some statistical sense. In a non-stationary environment, the algorithm offers a **tracking capability**, whereby it can track time variations in the statistics of the input data, provided that the variations are sufficiently slow.

Application of a recursive algorithm, whereby the parameters of an adaptive filter are updated from one iteration to the next, the parameters become data dependent. This, therefore, means that an adaptive filter is a **non-linear** device in the sense that it **does not obey the principle of superposition**. However, an adaptive filter is often referred to as linear in the sense that the estimate of a quantity of interest is obtained adaptively (at the output of the filter) as a linear combination of the available set of observations applied to the filter input.

The choice of a finite-duration impulse response **(FIR)** or an infinite duration impulse response **(IIR)** for the filter is dictated by practical considerations. The choice of a statistical criterion for optimizing the filter design is influenced by mathematical tractability.

An FIR filter is inherently stable, because its structure involves the use of forward paths only. In other words, the only mechanism for input-output interaction in the

filter is via forward paths from the filter input to its output. This form of signal transmission through the filter limits its impulse response to a finite duration.

An IIR filter involves both feed forward and feedback paths. The presence of feedback path means that portions of the filter output and possibly other internal variables in the filter are fed back to the input. Consequently, unless it is properly controlled, feedback in the filter can make it unstable with the result that the filter oscillates; this kind of operation is clearly unacceptable when the requirement is for stability. By itself, the stability problem in IIR filters is manageable in both theoretical and practical terms. However, when the filter is adaptive, bringing with it stability problems of its own, the inclusion of adaptivity combined with feedback that is inherently present in an IIR filter makes a difficult problem that much more difficult to handle. Hence in the majority of applications requiring the use of adaptively, the use of an FIR filter is preferred over an IIR filter even though the latter is less demanding in computational requirements.

3.1.2 Cost Functions

The filter design is optimized by minimizing a **cost function, or index of performance**, it can be:

1. Mean-square value of the estimation error
2. Expectation of the absolute value of the estimation error
3. Expectation of third or higher powers of the absolute value of the estimation error.

First has advantage over the others that it leads to tractable mathematics. In particular, the choice of the mean-square error criterion results in second-order dependence for the cost function on the unknown coefficients in the impulse response of the filter.

A coefficient vector (that is called a tap weight vector as in a transversal filter, or a vector of reflection coefficients as in a multistage lattice filter) characterizes a discrete time **filter**. A cost function defines a transformation from a vector space spanned by the elements of the coefficient vector into the space of a real scalar.

The mean square error criterion can work as cost function. It is defined as the mean square value of an estimation error, e (n). e (n) is defined by the difference between desired response d (n) and the actual filter output y (n),

$$e(n) = d(n) - y(n) \qquad \text{.....(3.1)}$$

Transversal filter is characterized by the tap – weight vector W. The filter output is given by the inner product of tap-weight vector w and the tap input vector x(n).

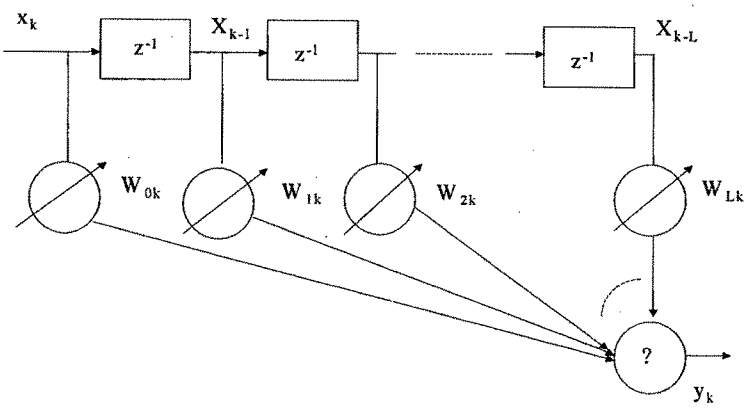


Figure3.1 Transversal filter

If u(n) and w are complex valued,

$$y(n) = w^H x(n) \qquad \text{.....(3.2)}$$

Where the superscript H indicates **Hermitian transposition** (i.e., conjugate transposition). The use of Hermitian transposition is preferred over ordinary transposition as it simplifies the appearance of the equations that define the characterization of optimum filters. A cost function for the optimum design of the transversal filter is

$$J(W) = E[|e(n)|^2]$$
$$= E[|d(n) - w^H x(n)|^2] \qquad \text{.....(3.3)}$$

Where E is the expectation operator. Finding the tap – weight vector that minimizes the cost function J, optimizes the filter. Such an approach is the minimum mean-square error criterion, **which forms the basis of Wiener filters.**

The cost function defined in Equation (3.3) is probabilistic in nature as it involves ensemble averaging represented by the expectation operator E.

Continuing with the example of a transversal filter, for given set of observations represented by the tap-input vector x(i), where the time index i = 1, 2 . . . , n. It is

assumed that the tap-weight vector w of the transversal filter is held constant for the entire observation interval. At time i, the estimation error is

$$e(i) = d(i) - w^H x(i) \quad \text{.....(3.4)}$$

Where d(i) is the corresponding value of the desired response. In this case, the cost function may be defined as

$$\xi(W,n) = \sum |e(i)|^2$$
$$= \sum |d(i) - w^H x(i)|^2 \quad \text{.....(3.5)}$$

Whereas the ensemble averaged cost function J(w) in Equation (3.3) depends only on the tap-weight vector w, the time-averaged cost function $\xi(W,n)$ depends on w and the observation interval n. Accordingly, the minimization $\xi(W,n)$ with respect to w yields a solution for the tap-weight vector that varies with the observation interval. This second approach is the basis of the **method of least squares.**

The cost function J(w) and $\xi(W,n)$ are both **convex with a unique minimum point.** Their use yields a unique solution for the tap-weight vector of the transversal filter.

Limitation of second-order statistics (e.g., the mean-square-error criterion) is that they are **phase blind.** This limitation can be overcome by the use of a **non-linear cost function.** By so doing, the filter is enabled to extract information (particularly phase information) from the input signal in a more efficient manner. For this to be possible, however, the input signal must have **non-Gaussian statistics.** The use of such an approach provides the basis for an important class of **nonlinear adaptive filtering algorithms** that can perform blind de-convolution.

3.1.3 Adaptive Filters

There is no unique solution to the adaptive filtering problem but there are a variety of **recursive algorithms,** each of which offers desirable features of its own [2]. Basically, there are **three** distinct methods for deriving recursive algorithms for the operation of adaptive filters: **Wiener Filter, Kalman Filter and Method of Least Squares**

A tapped delay line or transversal filter is used as the structural basis for implementing the adaptive filter. For the case of stationary inputs, the mean-squared error is a second-order function of the tap weights in the **transversal filter.** The dependence of the mean-squared error on the unknown tap weights is in the form of a

multidimensional paraboloid (i.e., punch bowl) with a uniquely defined bottom or minimum point. This paraboloid is known as the **error performance surface.** The tap weights corresponding to the minimum point of the surface define the **optimum Wiener solution.**

To develop a recursive algorithm for updating the tap weights of the **adaptive transversal filter**, there are two steps:

Step: 1 Modifying the system of Wiener Hop equations (i.e., the matrix equation defining the optimum Wiener solution) through the use of the method of **steepest descent.** It is a well-known technique in optimization theory.
This modification requires the use of a **gradient vector.** Its value depends on two parameters: The correlation matrix of the tap inputs in the transversal filter and cross-correlation vector between the desired response and the same tap inputs.

Step 2: Using instantaneous values for these correlations: Instantaneous values for these correlations are used to derive an estimate for the gradient vector. The resulting algorithm is known as the **least mean square (LMS)** algorithm, which is a **stochastic gradient algorithm.** It iterates each tap weight in the transversal filter in the direction of the gradient of the squared amplitude of an error signal with respect to that tap weight. It is similar to the stochastic approximation developed by Robbins and Monroe (Robbins and Monroe, 1951) the step size parameter is made inversely proportional to time n or to a power of n except that LMS uses a fixed step-size parameter to control the correction applied to the tap weight during the iterations. In a non-stationary environment, the orientation of the error-performance surface varies continuously. The LMS algorithm uses a **transversal filter structure**, whereas the gradient lattice algorithm (GAL) uses a **lattice structure.**

Classically, filter design problem can be stated as:
"Design a linear discrete time filter whose output y(n) provides an estimate of a desired response, given a set of input samples u(0), u(1), u(2) . . . such that the mean square value of the estimation error e(n), defined as the difference between the desired response d(n) and the actual response y(n), is minimized." is based on statistical theory of Wiener and Kalman filtering.

3.1.3.1 Method of least squares

It involves the use of time averages. The index of performance that consists of the sum of weighted error squares is minimized, where the error is defined as the difference between some desired response and the actual filter output. The method of

least squares can be formulated with **block estimation** or **recursive estimation.** In block estimation the input data stream is arranged in the form of blocks of equal length (duration), and the filtering of input data proceeds on a block-by-block basis. In recursive estimation, on the other hand, the estimates of interest (e.g., tap weights of a transversal filter) are updated on a sample-by-sample basis. Generally, a recursive estimator requires less storage than a block estimator. There are three basically different classes of adaptive filtering algorithms that originate from the method of least squares:

- **Recursive least squares algorithm:**

As with adaptive filtering algorithms derived from the Wiener and Kalman filters, the recursive least-squares (RLS) algorithm filter also assumes the use of transversal filters as the structural basis of the adaptive filter. The derivation of the algorithm relies on the matrix inversion lemma. The RLS algorithm is a **special case of the Kalman algorithm** for adaptive transversal filters. The Kalman or RLS algorithm usually provides a much faster rate of convergence than the LMS algorithm at the expense of increased computational complexity.

- **QR – decomposition based recursive least-squares (QRD-RLS) algorithm:**

This class of adaptive filtering algorithms is based on the QR-decomposition of the data matrix. Two well-known techniques for performing this decomposition are the **Householder transformation** and the **Givens rotation**. These are data adaptive transformations that involve orthogonal triangularization of the incoming data matrix. They are both highly popular in modern numerical analysis. At this point in the discussion, the important point to note is that a recursive least-squares algorithm based on the **Householder** transformation or Givens rotation is numerically stable and robust. Use of **Givens rotations** is that it is amenable to recursive computation, with the rotation parameters being updated on a sample-by-sample basis. The latter form of computation leads to the use of systolic arrays for implementing recursive least-squares estimation.

- **Fast algorithms:**

The standard RLS algorithm (using a transversal structure) and its counterpart based on the QR-decomposition (using a systolic array) have a computational complexity that increases as the square of M, where M is the number of adjustable weights which represent the number of degrees of freedom) in the algorithm. Such algorithms are O (M^2) algorithms. On the other hand, the LMS algorithm is an O(M) algorithm. When M is large, the computational complexity of the O (M^2) algorithms may be come objectionable from a hardware implementation point of view. There is therefore a

strong motivation to modify the formulation of recursive linear least-squares algorithms in such a way that the computational complexity assumes an O (M) form. This objective is achievable (i) by virtue of the inherent redundancy in the Toeplitz structure of the input data matrix, and (ii) by exploiting this redundancy through the use of linear least-squares estimation with an O(M) computational complexity. Three types of fast algorithms are identified, depending on the filter structure or technique employed:

a) **Fast transversal filters (FTF) algorithm:** This algorithm involves a parallel combination of four transversal filters, each one of which has an assigned task to perform.
b) **Recursive least squares lattice (LSL) algorithms:** relies on the use of a lattice structure to perform the forward and backward forms of linear least-squares prediction. There are four distinct forms of the recursive LSL algorithm.
c) **QR-decomposition based least squares lattice (QRD-LSL) algorithm:** The forward and backward forms of linear least-squares prediction are exploited for the purpose of reducing the computational complexity in performing the recursive QR-decomposition of the input data matrix.

The QR-decomposition based least-squares lattice (QRD-LSL) algorithm is numerically stable because of the good numerical properties inherently associated with the QR-decomposition method.

3.1.3.2 Method of Steepest Descent

The method of steepest descent is recursive. Starting from some initial (arbitrary) value for the tap-weight vector, it improves with the increased number of iterations. The final value so computed for the tap-weight vector converges to the **Wiener solution**. Deterministic control system finds the minimum point of the ensemble-averaged error-performance surface without knowledge of the surface itself. Accordingly, it provides heuristics for writing the recursions that describe the **least-mean-square (LMS) algorithm**.

The steps to find the minimum value of the mean-squared error, J_{min}, by the steepest descent algorithm are described below:

1) Begin with an initial arbitrary value w(0) for the tap weight vector to provide an initial guess as to where the minimum point of the error-performance surface may be located. Typically, w(0) is set equal to the null vector.

2) Using this initial or present guess, computer the gradient vector is computed, the real and imaginary parts of which are defined as the derivative of the mean squared error J(n), evaluated with respect to the real and imaginary parts of the tap-weight vector w(n) at time n(i.e., the nth iteration).
3) Computer the next guess at the tap weight vector by making a change in the initial or present guess in a direction opposite to that of the gradient vector.
4) Go back to step 2 and repeat the process.

3.1.3.3. Least-Mean-Square Algorithm:

The operation of the least-mean-square (LMS) algorithm is descriptive of a feedback control system. Basically, it consists of a combination of two basic processes:

1) An adaptive process, which involves the automatic adjustment of a set of tap weights.
2) A filtering process, which involves (a) forming the inner product of a set of tap inputs and the corresponding set of tap weights emerging from the adaptive process to produce an estimate of a desired response, and (b) generating an estimation error by comparing this estimate with the actual value of the desired response. The estimation error is in turn used to actuate the adaptive process, thereby closing the feedback loop.

For the convergence to hold, the step-size parameter μ has to satisfy different conditions related to the eigen values of the correlation matrix of the tap inputs.

The difference between the final value $J(\infty)$ and the minimum value Jmin is called the **excess mean squared error** $Jex(\infty)$. The ratio of $Jex(\infty)$ to Jmin is called the **mis-adjustment**, which is a measure of how far the steady state solution computed by the LMS algorithm is away from the Wiener solution.

3.2 Signal Compression

Signal compression is used to achieve a low bit rate in the digital representation of the signals with a minimum loss of the signal quality. Compression is usually referred to as low bit rate coding or coding, for short. Signal compression has found a wide use in many respect of signal communications, signal storage and message encryption. A good compromise involving quality, complexity and compression ratio has not yet been reached.

An electrocardiogram (ECG) data is very useful for cardiac disease diagnostic. Modern computerized ECG recording systems produce vast amount of sampled

heartbeat data. Storing sampled ECG data of many patients requires a huge storage capacity. For efficient storage of such large data records, effective data compression methods are of interest. The aim of ECG compression is to reduce the amount of digitized ECG data as much as possible with a reasonable implementation complexity while maintaining clinically acceptable signal quality.

The techniques compression techniques can be classified as: Direct methods and Transform methods.

- **Direct methods:** The compression is directly done on the ECG samples. Commonly used methods are: Amplitude Zone Time Epoch Coding (AZTEC), Turning Point (TP), Coordinate Reduction Time Encoding (CORTES), Scan Along Polynomial Approximation (SAPA), Peak peaking, cycle-to-cycle and Differential Pulse Code Modulation (DPCM).
- **Transform Methods:** The original samples are transformed to another domain in the hope of achieving better compression performance. Commonly used methods include: Fourier Descriptors, Walsh Transforms, Karhunen – Loève Transform (KLT), Discrete Cosine Transforms (DCT), and Wavelet Transform.

Direct methods are better than transform methods with respect to system complexity and the error control mechanism. Transform methods achieve higher compression ratios and are insensitive to presence of noise in the original ECG signals.

Since ECG are non-stationary, the error control problem for a reconstructed ECG signal is an important issue. In direct methods the error limit for a reproduced ECG signal is easily controlled by adjusting a user-specified error threshold. Since in transform techniques the distortion of each reconstructed segment of the ECG signal varies with the complex pattern of the segment and it is difficult to find a predetermined optimal quantization size for good quality reconstructed ECG signal at every segment the error control is difficult.

A compression algorithm can be evaluated in terms of relative complexity, memory requirement, speed of execution, and amount of compression and reconstruction error.

Performance of compression algorithm is measured as **compression ratio,** which is the ratio of the number of bits required to represent the data before compression to the number of bits required to represent the data after compression. It can also be

represented by expressing the reduction in the amount of data required as a percentage of the size of the original data or rate which is the average number of bits required to represent a single sample.

Difference between the original and the reconstruction is called the **distortion.** Because human responses are difficult to model mathematically, many approximate measures of distortion are used to determine the quality of the reconstructed waveforms.

3.2.1 Compression Techniques

Based on the requirements of reconstruction, data compression schemes is divided into two broad classes: (a) lossless compression schemes, in which Y(Reconstructed) is identical to X(Original), and (b) lossy compression schemes, which generally provide much higher compression than lossless compression but allow Y to be different from *X.*

- **Lossless compression** techniques involve no loss of information. If data have been losslessly compressed, the original data can be recovered exactly from the compressed data. Lossless compression is generally used for applications that cannot tolerate any difference between the original and reconstructed data. Text compression is an important area for lossless compression. It is very important that the reconstruction is identical to the text original, as very small differences can result in statements with very different meanings.
- **Lossy compression** techniques involve some loss of information, and data that have been compressed using lossy techniques generally cannot be recovered or reconstructed exactly. In return for accepting this distortion in the reconstruction, higher compression ratio is possible as compared to lossless compression.

In many applications, lack of exact reconstruction is not a problem. Depending on the quality required of the reconstructed speech, varying amounts of loss of information about the value of each sample can be tolerated.

3.2.2 Modeling and Coding

Compression scheme depends on a number of different factors. Some of the most important factors are the characteristics of the data that need to be compressed. A compression technique that will work well for the compression of text may not work well for compressing images. Each application presents a different set of challenges.

The development of data compression algorithms for a variety of data can be divided into two phases. The first phase is ***modeling***. In this phase information about any redundancy that exists in the data is extracted and the redundancy is described in the form of a model. The second phase is ***coding***. A description of the model and a description of how the data differ from the model are encoded, generally using a binary alphabet. The difference between the data and the model is called the ***residual.***

3.2.2.1 Huffman Coding Algorithm

It is a very popular coding algorithm. The codes generated using this technique or procedure is called **Huffman codes**. These codes are prefix codes and are optimum for a given model.

The Huffman procedure is based on two observations regarding optimum prefix codes.

1. In an optimum code, symbols that occur more frequently (have a higher probability of occurrence) will have shorter code words than symbols that occur less frequently.
2. In an optimum code, the two symbols that occur least frequently will have the same length.

The Huffman procedure is obtained by adding a simple requirement to these two observations. This requirement is that the code words corresponding to the two lowest probability symbols differ only in the last bit. That is, if γ and δ are the two least probable symbols in an alphabet, if the codeword for γ was **m * 0,** the codeword for δ would be **m * 1.** Here **m** is a string of 1s and 0s, and * denotes concatenation.

Applications of Huffman coding include lossless image compression, text compression, audio compression etc.

3.2.2.2. Arithmetic Coding

Arithmetic coding is especially useful when dealing with sources with small alphabets, such as binary sources, and alphabets with highly skewed probabilities. It is also a very useful approach when, for various reasons, the modeling and coding aspects of lossless compression is to be kept separate. A variation of the arithmetic code is the coding method used in the Joint Bi-level Experts Group (JBIG) standard for encoding binary images.

3.2.2.3 Dictionary Techniques

Dictionary techniques - both static and adaptive (or dynamic)-build a list of commonly occurring patterns and encode these patterns by transmitting their index in the list. They are most useful with sources that generate a relatively small number of patterns quite frequently, such as text sources and computer commands.

A very reasonable approach to encoding such sources is to keep a list, or dictionary, of frequently occurring patterns. When these patterns appear in the source output, they are encoded with a reference to the dictionary. This technique to be effective, the class of frequently occurring patterns, and hence the size of the dictionary, must be much smaller than the number of all possible patterns.

One of the more common forms of static dictionary coding is **diagram coding.** Where all letters of the source alphabet followed by as many pairs of letters, called digrams, as can be accommodated by the dictionary.

The digram encoder reads a two-character input and searches the dictionary to see if this input exists in the dictionary. If it does, the corresponding index is encoded and transmitted. If it does not, the first character of the pair is encoded. The second character in the pair then becomes the first character of the next digram. The encoder reads another character to complete the digram, and the search procedure is repeated.

3.2.2.4 Predictive Coding

These techniques make use of the past history of the data being encoded to provide more efficient compression. Number of schemes is principally used for the compression of text and images.

It is learned that we get more compression when the message that is being coded has a more skewed set of probabilities. There are several ways to achieve this situation. We can transform the sequence (in an invertible fashion) into another sequence that has the desired property. Or, we can use a different probability distribution for each symbol, such that in the probability distribution used with high likelihood the symbol being coded is a high-probability symbol. In both approaches we also need to let the decoder know the transformation or the distribution function being used. If the transformation or distribution is based on the history of the sequence, history that is available to both encoder and decoder, then there might not he any need to transmit the additional information. Because we use the history of the sequence in a predictive

manner to determine its encoding, such schemes are called **predictive coding schemes.**

A different kind of prediction is used when encoding non numerical data such as text. When encoding a particular symbol, one would like to use the probability distribution in which the symbol one is trying to encode has a high probability. One can then use adaptive arithmetic coding to encode that symbol at a low rate. However, if one is going to use a different probability model to encode each symbol, for each symbol one need to let the receiver know which probability model is being used. One way to do this with no cost is to use the preceding symbols to predict the probability model. In other words, the preceding symbols form the *context* in which the symbol is being encoded.

Some of the best performing text compression algorithms are variants of the prediction with partial match (ppm) algorithm. The new JPEG standard for lossless image compression is a predictive coding algorithm.

3.2.2.5 Scalar Quantization

In many lossy compression applications one is required to represent each source output using one of a small number of codeword. The number of possible distinct source output values is generally much larger than the number of codewords available to represent them. The process of representing a large-possibly infinite-set of values with a much smaller set is called quantization.

The set of inputs and outputs of a quantizer can be scalars or vectors. If they are scalars, they are called scalar quantizers. If they are vectors, they are called vector quantizers.

3.2.2.6 Vector Quantization

Grouping source outputs together and encoding them as a single block can obtain efficient lossy as well as lossless compression algorithms. Many of the lossless compression algorithms that are looked at take advantage of this fact. Blocks of data are also called vectors, hence called "vector Quntization."

Even when the input is random, encoding sequences of samples instead of encoding individual samples separately provides a more efficient code. Encoding sequences of samples is more advantageous in the lossy compression framework as well. By "advantageous" we mean a lower distortion for a given rate, or a lower rate for a given distortion.

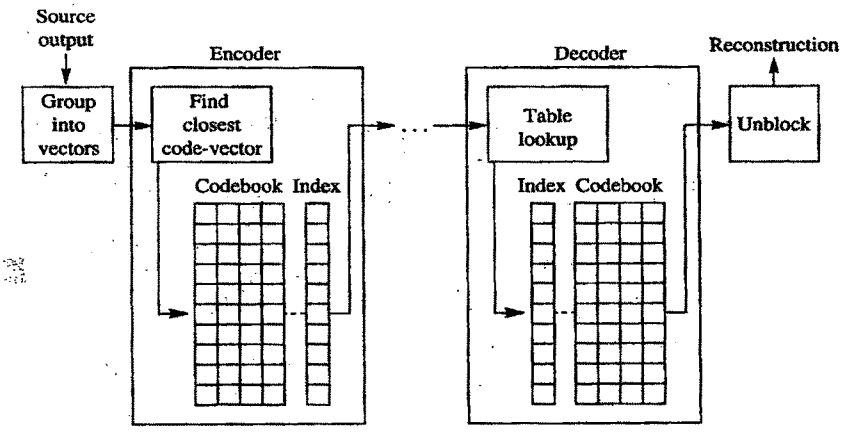


Figure 3.2 Vector Quantization

Consider a block of L pixels from an image and treat each pixel value as a component of a vector of size or dimension L. This vector of source outputs forms the input to the vector quantizer. A set of L-dimensional vectors at both the encoder and decoder of the vector quantizer are called the **codebook.** The vectors in this codebook, known as *code-vectors* are generated from the source output. Each code-vector is assigned a binary index. At the encoder, the input vector is compared to each code-vector to find the code-vector closest to the input vector. The elements of this code-vector are the quantized values of the source output. In order to inform the decoder about which code-vector was found to be the closest to the input vector, Binary index of the code-vector is transmitted or stored. As the decoder has exactly the same codebook, it can retrieve the code-vector given its binary index as in **Figure 3.2.**

Although the encoder may have to perform a considerable amount of computations in order to find the closest reproduction vector to the vector of source outputs, the decoding consists of a table lookup. This makes vector quantization a very attractive encoding scheme for applications in which the resources available for decoding are considerably less than the resources available for encoding. However, if the decoding is to be done in software, the amount of computational resources available to the decoder may be quite limited.

3.2.2.7 Differential Encoding

Sources such as speech and images have a great deal of correlation from sample to sample. We can use this fact to predict each sample based on its past and only encode

and transmit the differences between the prediction and the sample value. Differential encoding schemes are built around this premise. Because the prediction techniques are rather simple, these schemes are much easier to implement than other compression schemes.

3.2.2.8 Modified Huffman Coding

The implementation of Huffman coding requires a translation table, where each source symbol is mapped to unique code word. For example, if the original data were quantized into 16-bit numbers, the table would need to contain 2^{16} records. A table of this size creates memory problems and processing inefficiency.

In order to reduce the size of the translation table, the modified Huffman coding scheme partitions the source symbols into a frequent set and an infrequent set. For all the symbols in the frequent set, we form a Huffman code as in the static scheme. One then need to use a special code word as a prefix to indicate any symbol from the infrequent set and attach a suffix corresponding to the ordinary binary encoding of the symbol.

3.2.2.9 Adaptive Coding

Huffman coding requires a translation table for encoding and decoding. It is necessary to examine the entire data set or portions of it to determine the data statistics. The translation table must also be transmitted or stored for correct decoding.

An adaptive coding scheme attempts to build the translation table as data are presented. A dynamically derived translation table is sensitive to the variation in local statistical information. It can therefore alter its code words according to local statistics to maximize the reduction ration. It also achieves extra space saving because there is no need for a static table.

An example of an adaptive scheme is the Lempel-Ziv-Welch (LZW) algorithm. The LZW algorithm uses a fixed-size table. It initializes some positions of the table for some chosen data sets. When it encounters new data, it uses the uninitialized so that each unique data word is assigned its own position. When the table is full, the LZW algorithm reinitializes the oldest or least-used position according to the new data. During data reconstruction, it incrementally reconstruction, it incrementally rebuilds the translation table from the encoded data.

3.2.2.10 Residual differencing

Typically, neighboring signal amplitudes are not statistically independent. Conceptually one can decompose a sample value into a part that is correlated with past samples and a part that is uncorrelated. Since the inter sample correlation corresponds to a value predicted using past samples, it is redundant and removable. We are then left with the uncorrelated part which represents the prediction error or residual signal. Since the amplitude range of the residual signal is smaller than that of the original signal, it requires less bits for representation. One can further reduce the data by applying Huffman coding to the residual signal.

3.2.2.11 Run length encoding

It is used extensively in the facsimile technology, run-length encoding exploits the high degree of correlation that occurs in successive bits in the facsimile bit streams. A bit in the facsimile output may either be 1 or 0, depending on whether it is a black or white pixel. On a typical document, there are clusters of black and white pixels that give rise to this high correlation. Run-length encoding simply transforms the original bit stream into the string $\{v_1, l_1, v_2, l_2, \ldots\}$ where v_i are the values and l_i are lengths. The observant reader will quickly recognize that both AZTEC and the Fan algorithm are special cases of run-length encoding.

3.2.3 Biomedical Signal Compression Algorithms

The section describes compression algorithms for biomedical signals.

3.2.3.1 Turning point algorithm

The original motivation for the turning point (TP) algorithm was to reduce the sampling frequency of an ECG signal from 200 to 100 samples/s (Mueller, 1978). The algorithm developed from the observation that, except for QRS complexes with large amplitudes and slopes, a sampling rate of 100 samples/s is adequate.

TP is based on the concept that ECG signals are normally over sampled at four or five times faster that the highest frequency present. For example, an ECG used in monitoring may have a bandwidth of 50 Hz and be sampled at 200 sps in order to easily visualize the higher-frequency attributes of the QRS complex. Sampling the signal is done at reduced effective sampling rate (by half) of 100 sps by selectively saving important signal points (i.e., the peaks and valleys or turning points.)

The algorithm processes three data point at a time. It stores the first sample point and assigns it as the reference point X_0. The next two consecutive points become X_1 and

X_2. The algorithm retains either X_1 or X_2, depending on which point preserves the turning point (i.e., slope change) of the original signal.

The TP algorithm is simple and fast, producing a fixed reduction ratio of 2:1. After selectively discarding exactly half the sampled data, the original resolution can be restored by interpolating between pairs of saved data points.

A second application of the algorithm to the already reduced data increases the reduction ration to 4:1. Using data acquired at a 200-sps rate, this produces compressed data with a 50-sps effective sampling rate. If the bandwidth of the acquired ECG is 50 Hz, this approach violates sampling theory since the effective sampling rate is less than twice the highest frequency present in the signal. The resulting reconstructed signal typically has widened QRS complex and sharp edges that reduce its clinical acceptability. Another disadvantage of this algorithm is that the saved points do not represent equally spaced time intervals. This introduces short-term time distortion. However, this localized distortion is not visible when the reconstructed signal is viewed on the standard clinical monitors and paper recorders.

3.2.3.2 AZTEC algorithm

Originally developed to preprocess ECGs for rhythm analysis, the AZTEC (Amplitude Zone Time Epoch Coding) data reduction algorithm decomposed raw ECG sample points into plateaus and slopes (Cox et al., 1968). It provides a sequence of line segments that form a piecewise-linear approximation to the ECG.

The reconstruction process produces and ECG signal with step like quantization, which is not clinically acceptable. The AZTEC-encoded signal needs post processing with curve smoothing algorithm or a low-pass filter to remove its jagged appearance and produce more acceptable output.

3.2.3.3 CORTES algorithm

The CORTES (Coordinate Reduction Time Encoding System) algorithm is a hybrid of the TP and AZTEC algorithms (Abenstein and Tompkins, 1982; Tompkins and Webster, 1981). It attempts to exploit the strengths of each while sidestepping the weaknesses. CORTES uses AZTEC to discard clinically insignificant data in the isoelectric region with high frequency regions (QRS complexes). It executes the AZTEC and TP algorithms in parallel on the incoming ECG data.

Whenever an AZTEC line is produced, the CORTES algorithm decides, based on the length of the line, whether the AZTEC data or the TP data are to be saved. If the line

is longer than an empirically determined threshold, it saves the AZTEC line. Otherwise it saves the TP data points. Since TP is used to encode the QRS complexes, only AZTEC plateaus, not slopes, are implemented.

The CORTES algorithm reconstructs the signal by expanding the AZTEC plateaus and interpolation between each pair of the TP data points. It then applies parabolic smoothing to reduce discontinuities.

3.2.3.4 Fan algorithm

Originally used for ECG telemetry, the Fan algorithm draws lines between pairs of starting and ending points so that all intermediate samples are within some specified error tolerance, ε (Bohs and Barr, 1988). It starts by accepting the first sample X_0 as the non redundant permanent point. It functions as the origin and is also called the originating point. It then take the second sample X_1 and draw two slopes $\{U_1, L_1\}$, U_1 passes through the point $(X_0+X_1+\varepsilon)$, and L_1 passes through the point $(X_0-X_2-\varepsilon)$. If the third sample X_2 falls within the area bounded by the two slopes, it generates two new slopes $\{U_2, L2\}$ that pass through points $(X_0, X_2+\varepsilon)$ and $(X_0, X_2-\varepsilon)$. It compares the two pairs of slopes and retains the most converging (restrictive) slopes (i.e., $\{U_1, L_2\}$ in this example). Next it assigns the value of X_2 to X_1 and read the next sample into X_2. As a result, X_2 always holds the most recent sample and X_1 holds the sample immediately preceding X_2. It repeats the process by comparing X_2 to the values of the most convergent slopes. If it falls outside this area, we save the length of the line T and its final amplitude X_1 which then becomes the new originating point X_0, and the process begins anew. The sketch of the slopes drawn from the origination sample to future samples forms a set of radial lines similar to a fan, giving this algorithm its name.

3.3 Signal averaging

One predominant application area of signal averaging is in electroencephalography [3]. The EEG recorded from scalp electrodes is difficult to interpret in part because it consists of a summation of the activity of the billions of brain cells. It is impossible to deduce much about the activity of the visual or auditory parts of the brain from the EEG. However, if one stimulates a part of the brain with a flash of light or an acoustical click, an evoked response occurs in the region of the brain that processes information for the sensory system being stimulated. By summing the signals that are evoked immediately following many stimuli and dividing by the total number of stimuli, we obtain an averaged evoked response. This signal can reveal a great deal about the performance of a sensory system.

Signal averaging sums a set of time epochs of the signal together with the superimposed random noise. If the time epochs are properly aligned, the signal waveforms directly sum together. On the other hand, the uncorrelated noise averages out in time. Thus, the signal-to-noise ration (SNR) is improved.

Signal averaging is based on the following characteristics of the signal and the noise:

1. The signal waveform must be repetitive (although it does not have to be periodic). This means that the signal must occur more than once but not necessarily at regular intervals.
2. The noise must be random and uncorrelated with the signal. In this application, random means that the noise is not periodic and that it can only be described statistically (e.g. by its mean and variance).
3. The temporal position of each signal waveform must be accurately known.

It is the random nature of noise that makes signal averaging useful. Each time epoch (or sweep) is intentionally aligned with the previous epochs so that the digitized samples from the new epoch are added to the corresponding samples from the previous epochs. Thus the time-aligned repetitive signals S in each epoch are added directly together so that after four epochs, the signal amplitude is four times larger than for one epoch ($4S$). If the noise is random and has a mean of zero and an average RMS value N, the RMS value after four epochs is the square root of the sum of squres (i.e., $(4N^2)^{1/2}$ OR $2N$). In general after n repetitions the signal amplitude is mS and the noise amplitude is $(m)^{1/2}N$. Thus, the SNR improves as the ratio of m to $m^{1/2}$.
Signal averaging is a kind of digital filtering process. The Fourier transform of the transfer of the transfer function of an averager is composed of a series of discrete frequency components. Each of these components has the same spectral characteristic and amplitude. Because of the appearance of its amplitude response, this type of filter is called a **comb filter.**

The width of each tooth decreases as the number of sweeps repetitions increases. The desired signal has a frequency spectrum composed of discrete frequency components, a fundamental and harmonics. Noise, on the other hand, has continuous distribution. As the bandwidth of each of the teeth of the comb decreases, this filter more selectively passes the fundamental and harmonics of the signal while rejecting the random noise frequencies that fall between the comb teeth. The signal averager, therefore, passes the signal while rejecting the noise.

Figure 3.3 Shows the block diagram of a typical averager. To average a signal such as the cortical response to an auditory stimulus, the system is simulated (in this case, human subject) with an auditory click to the stimulus input. Simultaneously, a trigger is provided which is derived from the stimulus that enables the summation of the sampled data (in this case, the EEG evoked by the stimulus) with the previous responses (time epochs or sweeps) stored in the buffer. When the averager receives the trigger pulse, it samples the EEG waveform at the selected rate, digitizes the signal, and sums the samples with the contents of a memory location corresponding to that sample interval (in the buffer). The process continues, stepping through the memory addresses until all addresses have been sampled. The sweep is terminated at this point. A new sweep begins with the next trigger and the cycle repeats until the desired number of sweeps have been averaged. The result of the averaging process is stored in the buffer, which can then be displayed on a CRT as the averaged evoked response.

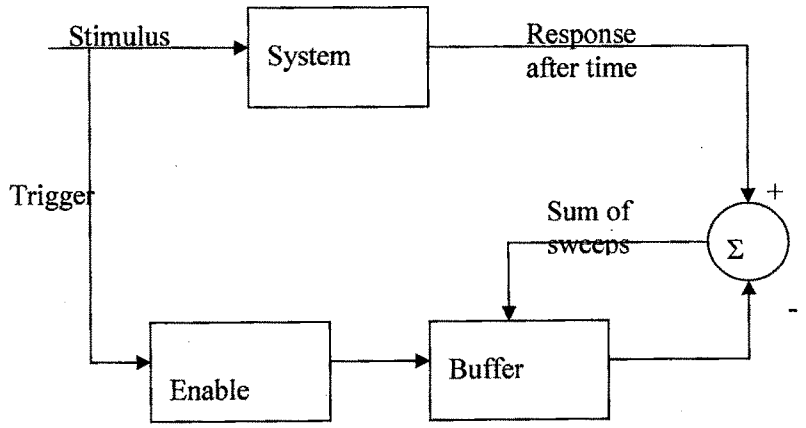


Figure 3.3 Block diagram of a typical signal averager.

An important assumption made in signal averaging theory is that the noise is Gaussian. This assumption is not usually completely valid for biomedical signals. Also, if the noise distribution is related to the signal, misleading results can occur. If the fiducial point is derived from the signal itself, care must be taken to ensure that noise is not influencing the temporal location of the fiducial point. Otherwise, slight misalignment of the signal waveforms will lead to a low-pass filtering effect in the final result.

3.4 Feature Extraction

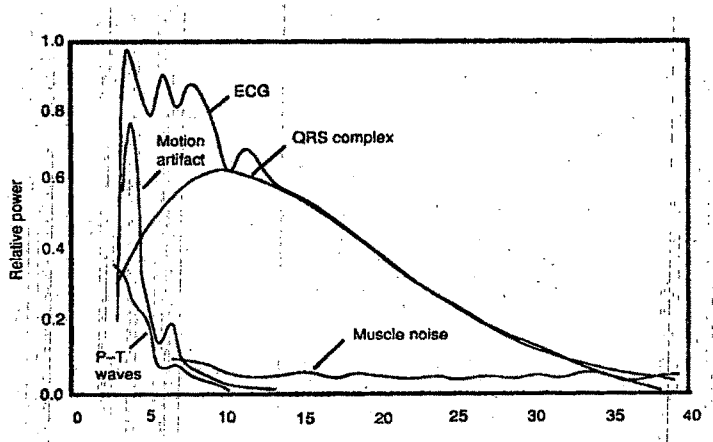
It is the process of studying and locating the areas and objects on the ground and deriving useful information from signals.

3.4.1 ECG: Power spectrum

The power spectrum of the ECG signal can provide useful information about the QRS complex. The power spectrum (based on the FFT) of a set of 512 sample points that contain approximately two heartbeats results in a series of coefficients with a maximal value near a frequency corresponding to the heart rate.

The heart rate can be determined by multiplying together the normalized frequency and the sampling frequency. One can also get useful information about the frequency spectrum of the QRS complex. In order to obtain this information, the QRS complex of the ECG signal must be selected as a template and zero-padded prior to the power spectrum analysis. The peak of the frequency spectrum obtained corresponds to the peak energy of the QRS complex.

The ECG waveform contains, in addition to the QRS complex, P and T waves, 60-Hz noise from power line interference, EMG from muscles, motion artifact from the electrode and skin interface, and possibly other interference from electro surgery equipment in the operating room. Many clinical instruments such as a cardio tachometer and arrhythmia monitor require accurate real-time QRS detection. It is necessary to extract the signal of interest, the QRS complex, from the other noise sources such as the P and T waves. **Figure 3.4** summarizes the relative power spectra of the ECG, QRS complexes, P and T waves, motion artifact, and muscle noise based on our previous research.



3.4 Relative power spectra of QRS complex, P and T waves, muscle noise and motion artifacts based on an average of 150 beats.

3.4.2 Band pass filtering

From the power spectral analysis of the various signal components in the ECG signal, a filter can be designed which effectively selects the QRS complex from the ECG. Another study that we performed examined the spectral plots of the ECG and the QRS complex from 3875 beats. **Figure 3.5** shows a plot of the signal-to-noise ratio (SNR) as a function of frequency. The study of the power spectra of the ECG signal, QRS complex, and other noise also revealed that a maximum SNR value is obtained for a band pass filter with a center frequency of 17 Hz and a Q of 3.

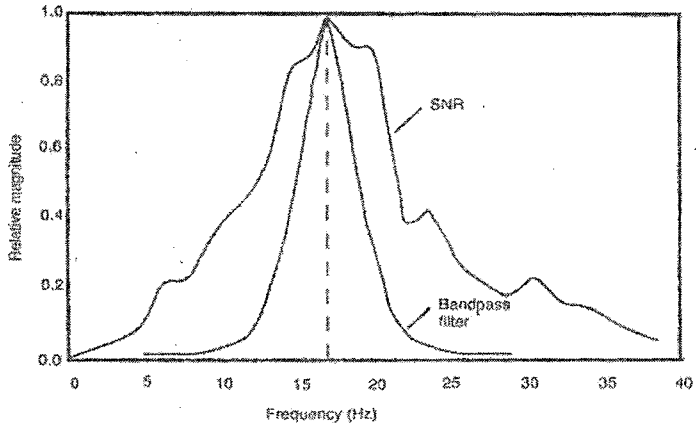


Figure 3.5 Plots of the signal-to-noise ratio (SNR) of the QRS complex referenced to all other signal noise based on 3875 heartbeats. The optimal bandpass filter for a cardio tachometer maximizes the SNR.

3.4.3 Two pole recursive filter

A simple two-pole recursive filter can be implemented to bandpass the ECG signal. The difference equation for the Filter is

$$y(nT) = 1.875y(nT-T) - 0.9219y(nT-2T) + x(nT) - x(nT-2T) \quad \text{.....(3.6)}$$

This filter design assumes that the ECG signal is sampled at 500 samples/s. The values of 1.875 and 0.9219 are approximations of the actual design values of 1.87635 and 0.9216 respectively. Since the coefficients are represented as powers of two, the multiplication operations can be implemented relatively fast using the shift operators.

3.4.4 Integer filter

QRS detectors for cardio tachometer application frequently band pass the ECG signal using a center using a center frequency of 17 Hz. The denominator of the general form of the transfer function allows for poles at 60^0, 90^0 and 120^0, and these

correspond to center frequencies of a band pass filter of T/6, T/4, and T/3 Hz, respectively. The desired center frequency can thus be obtained by choosing an appropriate sampling frequency.

3.4.5 Differentiation techniques

Differentiation forms the basis of many ORS detection algorithms. Since it is basically a high-pass filter, the derivative amplifies the higher frequencies characteristic of the QRS complex while attenuating the lower frequencies of the P and T waves.

An algorithm based on first and second derivatives originally developed by Balda et al. (1977) was modified for use in high-speed analysis of recorded ECGs by Ahlstrom and Tompkins (1983). Friesen et al. (1990) subsequently implemented the algorithm as part of a study to compare noise sensitivity among certain types of QRS detection algorithms.

3.4.6 Template matching techniques

In this section techniques for classifying patterns in the ECG signal that are quite related to the human recognition process are described.

3.4.6 .1 Template cross correlation

Signals are said to be correlated if the shapes of the waveforms of two signals match one another. The correlation coefficient is a value that determines the degree of match between the shapes of two or more signals. A QRS detection technique designed by Dobbs et al. (1984) uses cross correlation.

This technique of correlating one signal with another requires that the two signals be aligned with one another. In this QRS detection technique, the template of the signal that one is trying to match stores a digitized form of the incoming signal; the signal should be aligned with the template. Dobbs et al. describes two ways of implementing this:

The first way of aligning the template and the incoming signal is by using the fiducial point on each signal. This fiducial point has to be assigned to the signal by some external process. If the fiducial points on the template and signal are aligned, then the correlation can be performed.

Another implementation involves continuous correlation between a segment of the incoming signal and the template. Whenever a new signal data point arrives, the

oldest data point in time is discarded from the segment (a first-in-first-out data structure). A correlation is performed between this signal segment and the template segment that has the same number of signal points. This technique does not require processing time to assign fiducial points to the signal. The template can be thought of as a window that moves over the incoming signal one data point at a time. Thus, alignment of the segment of the signal of interest must occur at least once as the window moves though the signal.

The value of the cross correlation coefficient always falls between +1 and –1. A value of +1 indicates that the signal and the template match exactly. As mentioned earlier, the value of this coefficient determines how well the shapes of the two waveforms under consideration match. The magnitude of the actual signal samples does not matter. This shape matching, or recognizing process of QRS complexes, conforms with our natural approach to recognizing signals.

3.4.6 .2 Template subtraction

This is a relatively simple QRS detection technique. The algorithm begins by saving a segment of the incoming ECG signal that corresponds to the QRS waveforms. This segment or template is then compared with the incoming ECG signal at each point in the template. When the template is aligned with a QRS waveform in the signal, the subtraction results in a value very close to zero. This algorithm uses only as many subtraction operations as there are points in the template.

3.4.6 .3 Automata-based template matching

Furno and Tompkins (1982) developed a QRS detector that is based on concepts from automata theory. The algorithm uses some of the basic techniques that are common in many pattern recognition systems. The ECG signal is first reduced into a set of predefined tokens, which represent certain shapes of the ECG waveform.

3.4.6.4 A QRS detection algorithm

Over the past few years, there has been an increased trend toward processing of the electrocardiogram (ECG) using microcomputers. A survey of literature in this research area indicates that systems based on microcomputers can perform needed medical services in an extremely efficient manner. In fact, many systems have already been designed and implemented to perform signal processing tasks such as 12-lead off-line ECG analysis, Holter tape analysis, and real-time patient monitoring. All these applications require and accurate detection of the QRS complex of the ECG. For Example, arrhythmia monitors for ambulatory patients analyze the ECG in real time (Pan and Tompkins, 1985), and when an arrhythmia occurs, the monitor stores a time

segment of the abnormal ECG. This kind of monitor requires an accurate QRS recognition capability. Thus, QRS detection is an important part of many ECG signal processing systems.

A real-time QRS detection algorithm developed by Pan and Tompkins (1985) was further described by Hamilton and Tompkins (1986). It recognizes QRS complexes base on analyses of the slope, amplitude and width.

Figure 3.6 shows the various filters involved in the analysis of the ECG signal. In order to attenuate noise, the signal is passed through a band pass filter composed of cascaded high-pass and low-pass integer filters. Subsequent processes are differentiation, squaring and time averaging of the signal.

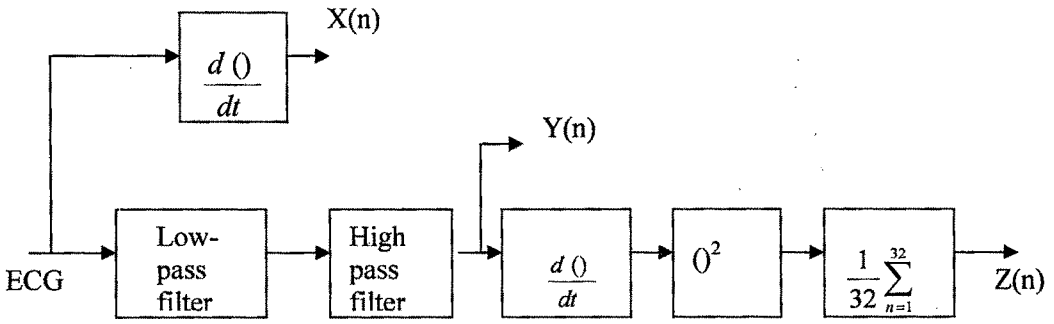


Figure 3.6 Filter stages of the QRS detector. Z (n) is the time-averaged signal. Y (n) is the band passed ECG, and x (n) is the differentiated ECG.