

# ECG Compression by Efficient Coding

Manjari Sharma

Student, M.E, MAC, Electrical Engineering Department,  
Madhav Institute of Technology and Science,  
Gwalior, M.P., India.  
manjari.sharma1@gmail.com

Dr. A. K. Wadhvani

Professor, Electrical Engineering Department,  
Madhav Institute of Technology and Science,  
Gwalior, M.P., India.  
wadhvani\_arun@rediffmail.com

**Abstract**—Electrocardiogram (ECG) data compression algorithm is needed to reduce the amount of data to be transmitted, stored and analyzed, without losing the clinical information content. This work investigates a set of ECG data compression schemes to compare their performances in compressing ECG signals. These schemes are based on transform methods such as discrete cosine transform (DCT), fast fourier transform (FFT), discrete sine transform (DST), and their improvements. An improvement of a discrete cosine transform (DCT)-based method for electrocardiogram (ECG) compression is also presented as DCT-II. A comparative study of performance of different transforms is made in terms of Compression Ratio (CR) and Percent root mean square difference (PRD). The appropriate use of a block based DCT associated to a uniform scalar dead zone quantiser and arithmetic coding show very good results, confirming that the proposed strategy exhibits competitive performances compared with the most popular compressors used for ECG compression. Each specific transform is applied to a pre-selected data segment from the CSE database, France and then compression is performed.

**Keywords**— ECG, compression, transform methods, CR, PRD.

## I. Introduction

Compression connotes the process of starting with a source of data in digital form (usually either a data stream or a stored file) and creating a representation that uses fewer bits than the original [1]. The aim is to reduce storage requirements or transmission time when such information is communicated over a distance. Ideally one needs the compression process to be reversible. Suppose a discrete time signal  $s(n)$  is compressed and then reconstructed, that is, the inverse of the compression process is performed to yield  $s^{\wedge}(n)$ . The error signal is defined as in (1):

$$e(n) = s(n) - s^{\wedge}(n) \quad (1)$$

The reconstructed signal can be alternatively taken as an additive noise contaminated version of the original by rewriting the above equation as in (2):

$$s^{\wedge}(n) = s(n) + w(n) \quad (2)$$

where  $w(n) = -e(n)$  is the noise. For a loss less compression  $e(n)$  is identically zero [1].

A typical computerized signal processing system acquires a large amount of data that is difficult to store and transmit. Data compression is nothing more-or-less than effective coding designed to correct the over representation that occurs

in digital data handling systems. The main aim of data compression is (a) To increase the storage efficiency. (b) Transmission bandwidth conservation. (c) Reducing the transmission time. The main goal of any compression technique is to achieve maximum data volume reduction while preserving the significant features [2] and also detecting and eliminating redundancies in a given data set.

ECG data compression algorithms have been mainly classified into three major categories [3]: 1) Direct time-domain techniques, e.g., turning point (TP), amplitude-zone-time epoch coding (AZTEC) [4], coordinate reduction time encoding system (CORTES) and Fan algorithm. 2) Transformational approaches [3], e.g., discrete cosines transformation (DCT), fast fourier transform (FFT), discrete sine transform (DST), wavelet transform (WT) etc. 3) Parameter extraction techniques, e.g., Prediction and Vector Quantization (VQ) methods [2].

Direct data compression methods rely on prediction or interpolation algorithms, which try to diminish redundancy in a sequence of data by looking at successive neighboring samples. Prediction algorithms employ a prior knowledge of previous samples, whereas interpolation algorithms use a prior knowledge of both previous and future samples. The direct data compression methods base their detection of redundancies on direct analysis of actual sample. Direct signal compression methods are also known as time domain techniques dedicated to compression of signals. The mode of operation is to extract a subset of significant samples from the original sample set. Which samples are significant, depends on the underlying criterion for the sample selection process. These algorithms suffer from sensitiveness to sampling rate, quantization levels and high frequency interference. It fails to achieve high data rate along with preservation of clinical information [5].

In Transform based techniques [6] compressions are accomplished by applying an invertible orthogonal transform to the signal. Due to its de-correlation and energy compaction properties the transform based methods achieve better compression ratios [7]. Transformation methods involve processing of the input signal by a linear orthogonal transformation and encoding of output using an appropriate error criterion. For signal reconstruction an inverse transformation is carried out and the signal is recovered with some error. Various orthogonal transformations include DCT, DST, FFT and WAVELET transforms etc.

The parameter extraction method includes extracting a particular parameter of the signal. The extracted parameters are subsequently utilized for classification based on a prior knowledge of the signal features. Direct and transformation methods are reversible, while parameter extraction method is irreversible.

In this work, ECG signal is compressed using the Discrete Cosine Transform (DCT), Fast Fourier Transform (FFT), Discrete Sine Transform (DST), and Type 2 Discrete Cosine Transform (DCT-II).

## II. need for ecg signal compression

The need for ECG compression exists in many transmitting and storage applications. Transmitting the ECG signal through telephone lines, for example, may save a crucial time and unnecessary difficulties in emergency cases. Effective storage is required of large quantities of ECG information in the intensive coronary care unit, or in Long-term (24–48 hours) wearable monitoring tasks (Holter). Holter monitoring usually requires continuous 12 or 24-hours ambulatory recording. For good diagnostic quality, each ECG lead should be sampled at a rate of 250–500 Hz with 12 bits resolution. The information rate is thus 11–22 Mbits/hour/lead approximately. The monitoring device must have a memory capacity of about 100–200 Mbytes for a 3-lead recording. Memory costs may render such a solid state Holter device impractical. In practice, efficient data compression may be achieved only with lossy compression techniques (which allow reconstruction error). In ECG signal compression algorithms the goal is to achieve a minimum information rate, while retaining the relevant diagnostic information in the reconstructed signal. All ECG compression algorithms have used simple mathematical distortion measures such as the percentage rms difference (PRD) for evaluating the reconstructed signal. It is used to evaluate the compression result.

## III. performance evaluation

Any performance criterion used to evaluate an ECG compression algorithm must include two factors Compression ratio, CR and Percent root mean square difference, PRD. In present work we have tested the data on the basis of these two.

### A. Compression Ratio (CR)

This is one of the most important parameters in data compression algorithms which specifies the amount of compression. All data compression algorithms minimize data storage by reducing the redundancy wherever possible, thereby increasing the compression ratio [7]. The compression ratio (CR) is defined as the ratio of the number of bits representing the original signal to the number of bits required to store the compressed signal. A high compression ratio is typically desired [2]. A data compression algorithm must also represent the data with acceptable fidelity while achieving high CR, given by (3).

$$CR = \frac{\text{TCB number of samples before compression}}{\text{TCB number of samples after compression}} \quad (3)$$

## B. Error Criteria and Distortion Methods

One of the most difficult problems in ECG compression applications and reconstruction is defining the error criterion. The purpose of the compression system is to remove redundancy and irrelevant information. Consequently the error criterion has to be defined so that it will measure the ability of the reconstructed signal to preserve the relevant information. Since ECG signals generally are compressed with lossy compression algorithms, we have to have a way of quantifying the difference between the original and the reconstructed signal, often called distortion. Different objective error measures namely; root mean square error (RMSE), percentage root mean difference (PRD), signal to noise ratio (SNR) are used for calculation of reconstruction error. The most prominently used distortion measure is the Percent Root mean square Difference (PRD) [8] that is given by (4)

$$PRD = \left[ \frac{\sum_{n=1}^{L_b} [x(n) - x'(n)]^2}{\sum_{n=1}^{L_b} [x(n)]^2} \right]^{(1/2)} \quad (4)$$

where  $x(n)$  is the original signal,  $x'(n)$  is the reconstructed signal and  $L_b$  is the length of the block or sequence over which PRD is calculated. PRD provides a numerical measure of the residual root mean square (rms) error.

## IV. transformation methods

In this work we have compared the performance of four different transformation methods for ECG compression and then their performance is evaluated. The various compression techniques have been discussed below:

### A. Discrete Cosine Transform (DCT)

A discrete cosine transform (DCT) expresses a sequence of finitely many data points in terms of a sum of cosine functions oscillating at different frequencies [9]. DCTs are important to numerous applications in science and engineering, from lossy compression of audio (e.g. MP3) and images (e.g. JPEG) (where small high-frequency components can be discarded), to spectral methods for the numerical solution of partial differential equations. The use of cosine rather than sine functions is critical in these applications. For compression, it turns out that cosine functions are much more efficient whereas for differential equations the cosines express a particular choice of boundary conditions [10]. In particular, a DCT is a Fourier-related transform similar to the discrete Fourier transform (DFT), but using only real numbers. DCTs are equivalent to DFTs of roughly twice the length, operating on real data with even symmetry (since the Fourier transform of a real and even function is real and even), where in some variants the input and/or output data are shifted by half a sample.

Discrete Cosine Transform is a basis for many signal and image compression algorithms due to its high de-correlation and energy compaction property [7]. A discrete Cosine Transform of  $N$  sample is defined as in (5):

$$F(u) = \sqrt{\frac{2}{N}} C(u) \sum_{x=0}^{N-1} f(x) \cos \left[ \frac{\pi(2x+1)u}{2N} \right] \quad (5)$$

$$u=0,1,\dots,N-1,$$

where  $C(u) = \frac{1}{\sqrt{2}}$  for  $u=0$   
 $=1$  otherwise.

The function  $f(x)$  represents the value of  $x$ th samples of input signals [7].  $F(u)$  represents a DCT coefficients. The inverse DCT is defined in similar fashion as in (6):

$$f(x) = \sqrt{\frac{2}{N}} \sum_{u=0}^{N-1} C(u) F(u) \cos \left[ \frac{\pi(2x+1)u}{2N} \right] \quad (6)$$

$$x=0, 1, \dots N-1.$$

### B. Fast Fourier Transform (FFT)

A fast Fourier transform (FFT) [11] is an efficient algorithm to compute the discrete Fourier transform (DFT) and its inverse [12]. There are many distinct FFT algorithms involving a wide range of mathematics, from simple complex-number arithmetic to group theory and number theory. A DFT decomposes a sequence of values into components of different frequencies but computing it directly from the definition is often too slow to be practical. An FFT is a way to compute the same result more quickly. Computing a DFT of  $N$  points in the naive way, using the definition, takes  $O(N^2)$  arithmetical operations [13], while an FFT can compute the same result in only  $O(N \log N)$  operations. The difference in speed can be substantial, especially for long data sets where  $N$  may be in the thousands or millions—in practice, the computation time can be reduced by several orders of magnitude in such cases, and the improvement is roughly proportional to  $N / \log(N)$ . This huge improvement made many DFT-based algorithms practical; FFTs are of great importance to a wide variety of applications, from digital signal processing and solving partial differential equations to algorithms for quick multiplication of large integers. The most well known FFT algorithms depend upon the factorization of  $N$ , but there are FFTs with  $O(N \log N)$  complexity for all  $N$ , even for prime  $N$ . Many FFT algorithms only depend on the fact that is an  $N$ th primitive root of unity, and thus can be applied to analogous transforms over any finite field, such as number-theoretic transforms.

Fast Fourier Transform is a fundamental transform in digital signal processing with applications in frequency analysis, signal processing etc [7]. The periodicity and symmetry properties of DFT are useful for compression. The  $u^{th}$  FFT coefficient of length  $N$  sequence  $\{f(x)\}$  is defined as in (7):

$$F(u) = \sum_{x=0}^{N-1} f(x) e^{-j2\pi ux/N} \quad (7)$$

$$u=0,1,\dots,N-1$$

And its inverse transform is calculated from (8):

$$f(x) = \frac{1}{N} \sum_{u=0}^{N-1} F(u) e^{j2\pi ux/N} \quad (8)$$

$$x=0,1,\dots,N-1$$

### C. Discrete Sine Transform (DST)

Discrete sine transform (DST) [14] is a Fourier-related transform similar to the discrete Fourier transform (DFT), but using a purely real matrix. It is equivalent to the imaginary parts of a DFT of roughly twice the length, operating on real data with odd symmetry (since the Fourier transform of a real and odd function is imaginary and odd), where in some variants the input and/or output data are shifted by half a sample. Like any Fourier-related transform, discrete sine transforms (DSTs) express a function or a signal in terms of a sum of sinusoids with different frequencies and amplitudes. Like the discrete Fourier transform (DFT), a DST operates on a function at a finite number of discrete data points. The obvious distinction between a DST and a DFT is that the former uses only sine functions, while the latter uses both cosines and sines (in the form of complex exponentials). However, this visible difference is merely a consequence of a deeper distinction: a DST implies different boundary conditions than the DFT or other related transforms [15].

Formally, the discrete sine transform is a linear, invertible function  $F: \mathbb{R}^N \rightarrow \mathbb{R}^N$  (where  $\mathbb{R}$  denotes the set of real numbers), or equivalently an  $N \times N$  square matrix. There are several variants of the DST with slightly modified definitions. The  $N$  real numbers  $x_0, \dots, x_{N-1}$  are transformed into the  $N$  real numbers  $X_0, \dots, X_{N-1}$  according to (9):

$$X_k = \sum_{n=0}^{N-1} x_n \sin \left[ \frac{\pi}{N+1} (n+1)(k+1) \right] \quad (9)$$

$$k=0, \dots, N-1$$

### D. Discrete Cosine Transform-II (DCT-II)

The most common variant of discrete cosine transform is the type-II DCT [16]. The DCT-II is typically defined as a real, orthogonal (unitary), linear transformation by the formula in (10):

$$C_k^{II} = \sqrt{\frac{2-\delta_{k,0}}{N}} \sum_{n=0}^{N-1} x_n \cos \left[ \frac{\pi}{N} \left( n + \frac{1}{2} \right) k \right], \quad (10)$$

for  $N$  inputs  $x_n$  and  $N$  outputs  $C_k^{II}$ , where  $\delta_{k,0}$  is the Kronecker delta ( $= 1$  for  $k=0$  and  $= 0$  otherwise).

DCT-II can be viewed as special case of the discrete Fourier transform (DFT) with real inputs of certain symmetry [17]. This viewpoint is fruitful because it means that any FFT algorithm for the DFT leads immediately to a corresponding fast algorithm for the DCT-II simply by discarding the redundant operations.

The discrete Fourier transform of size  $N$  is defined by (11):

$$X_k = \sum_{n=0}^{N-1} x_n \omega_N^{kn} \quad (11)$$

where  $\omega_N = e^{-\frac{2\pi i}{N}}$  is an  $N$ th primitive root of unity. In order to relate this to the DCT-II, it is convenient to choose a different normalization for the latter transform [17] as in (12):

$$C_k = 2 \sum_{n=0}^{N-1} x_n \cos \left[ \frac{\pi}{N} \left( n + \frac{1}{2} \right) k \right] \quad (12)$$

This normalization is not unitary, but it is more directly related to the DFT and therefore more convenient for the development of algorithms. Of course, any fast algorithm for  $C_k$  trivially yields a fast algorithm for  $C_k^H$  although the exact count of required multiplications depends on the normalization. In order to derive  $C_k$  from the DFT formula, we can use the identity  $2\cos(\pi l/N) = \omega_{4N}^{2l} + \omega_{4N}^{4N-2l}$  to write

$$\begin{aligned} C_k &= 2 \sum_{n=0}^{N-1} x_n \cos \left[ \frac{\pi}{N} \left( n + \frac{1}{2} \right) k \right] \\ &= \sum_{n=0}^{N-1} x_n \omega_{4N}^{(2n+1)k} + \sum_{n=0}^{N-1} x_n \omega_{4N}^{(4N-2n-1)k} \\ &= \sum_{n=0}^{4N-1} \widehat{x}_n \omega_{4N}^{nk} \end{aligned} \quad (13)$$

where  $\widehat{x}_n$  is a real-even sequence of length  $4N$ , defined as follows for  $0 \leq n < N$ :

$$\widehat{x}_{2n} = \widehat{x}_{4N-(2n+1)} = x_n \quad (14)$$

Thus, the DCT-II of size  $N$  is precisely a DFT of size  $4N$ , of real-even inputs, where the even-indexed inputs are zero.

### v. the proposed algorithms

The various compression techniques DCT, FFT, DST and DCT-II algorithms are compared with PRD and Compression ratio CR and best suitable is considered. The algorithms are performed on CSE database shown in Fig. 1.

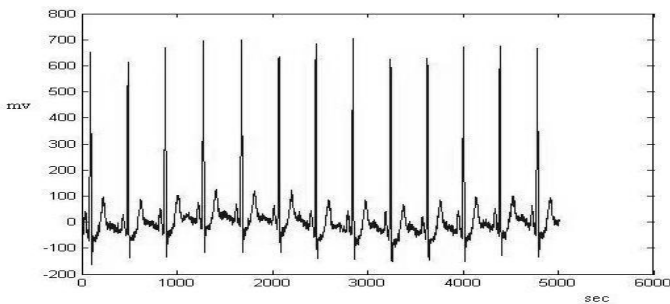


Figure 1. The ECG signal before compression

#### A. The DCT compression algorithm

- Separate the ECG components into three components  $x, y, z$ .
- Find the frequency and time between two samples.
- Find the DCT of ECG signal and check for DCT coefficients (before compression)  $=0$ , increment the counter A if it is between  $+0.22$  to  $-0.22$  and assign to  $\text{Index}=0$ .
- Check for DCT coefficients (after compression)  $=0$ , increment the Counter B.
- Calculate inverse DCT and plot decompression, error.

- Calculate the compression ratio CR and PRD.

The plot is shown in Fig. 2.

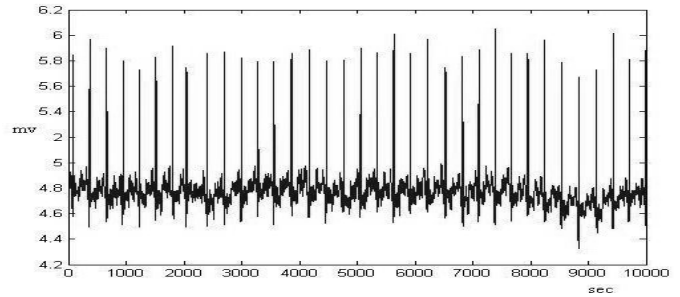


Figure 2(a). The ECG signal after DCT compression

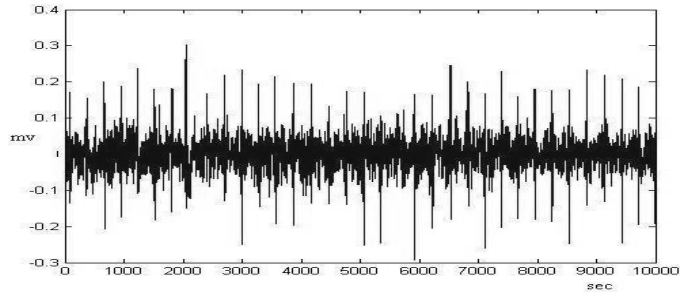


Figure 2(b). The Error signal after DCT compression

Figure 2. DCT compression analysis

#### B. The FFT compression algorithm

- Separate the ECG components into three components  $x, y, z$ .
- Find the frequency and time between two samples.
- Find the FFT of ECG signal and check for FFT coefficients (before compression)  $=0$ , increment the counter A if it is between  $+25$  to  $-25$  and assign to  $\text{Index}=0$ .
- Check for FFT coefficients (after compression)  $=0$ , increment the Counter B.
- Calculate inverse FFT and plot decompression, error.
- Calculate the compression ratio CR and PRD.

The plot is shown in Fig. 3.

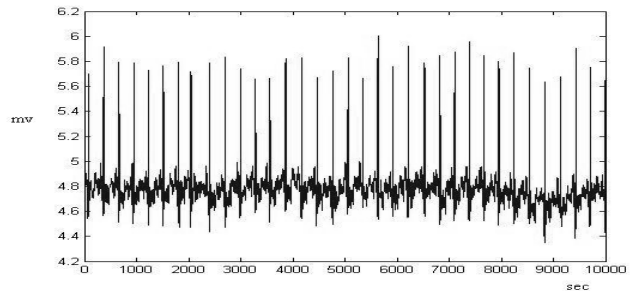


Figure 3(a). The ECG signal after FFT compression

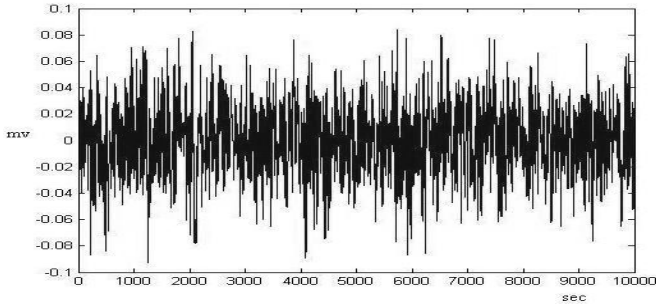


Figure 3(b). The Error signal after FFT compression

Figure 3. FFT compression analysis

### C. The DST compression algorithm

- Separate the ECG components into three components  $x, y, z$ .
- Find the frequency and time between two samples.
- Find the DST of ECG signal and check for DST coefficients (before compression)  $=0$ , increment the counter A if it is between  $+15$  to  $-15$  and assign to Index $=0$ .
- Check for DST coefficients (after compression)  $=0$ , increment the Counter B.
- Calculate inverse DST and plot decompression, error.
- Calculate the compression ratio CR and PRD.

The plot is shown in Fig. 4.

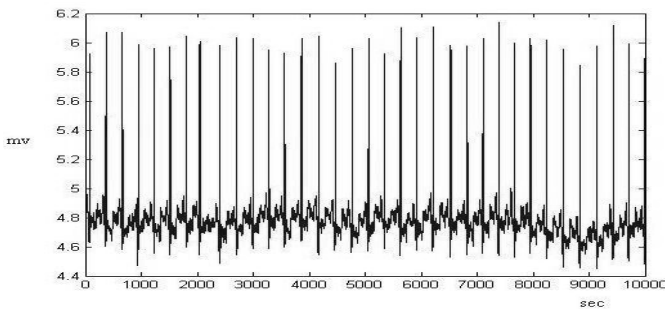


Figure 4(a). The ECG signal after DST compression

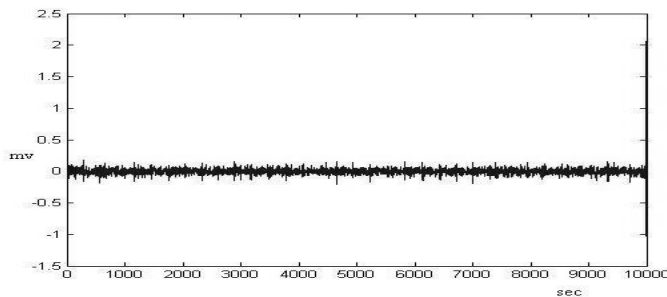


Figure 4(b). The Error signal after DST compression

Figure 4. DST compression analysis

### D. The DCT-II compression algorithm

- Partition of data sequence  $x$  in  $N_b$  consecutive blocks  $b_i, i=0, 1, \dots, N_b-1$ , each one with  $L_b$  samples.
- DCT computation for each block.
- Quantization of the DCT coefficients.
- Lossless encoding of the quantized DCT coefficients.

Increasing the block size increases the CR and the DCT computing time. Various results show that increasing the block size above a certain point results in a very modest CR gain, while the processing time increases. The type II DCT is commonly used for data compression due to its greater capacity to concentrate the signal energy in few transform coefficients.

The algorithm is explained in detail as:

- Let  $b_i[n], n=0, 1, \dots, L_b-1$ , represent the  $L_b$  values in block  $b_i$ .
- The one-dimensional DCT-II of this block generates a transformed block  $B_i$  constituted by a sequence of  $L_b$  coefficients  $B_i[m], m=0, 1, \dots, L_b-1$ , given by (15):

$$B_i[m] = \sqrt{\left(\frac{2}{L_b}\right)} c_m \sum_{n=0}^{L_b-1} b_i[n] \cos \left[ \frac{(2n+1)m\pi}{2L_b} \right] \quad (15)$$

$$m = 0, 1, \dots, L_b - 1$$

where  $c_m=1$  for  $1 \leq m \leq L_b-1$  and  $c_0=(1/2)^{(1/2)}$ .

The DCT can be seen as a one-to-one mapping for  $N$  point vectors between the time and the frequency domains. The coefficient  $B_i[0]$ , which is directly related to the average value of the time-domain block is called DC coefficient and the remaining coefficients of a block are called AC coefficients.

Given  $B_i$ ,  $b_i$  can be recovered by applying the inverse DCT-II:

$$b_i[n] = \sqrt{\left(\frac{2}{L_b}\right)} \sum_{m=0}^{L_b-1} c_m B_i[m] \cos \left[ \frac{(2n+1)m\pi}{2L_b} \right] \quad (16)$$

$$n=0, 1, \dots, L_b-1$$

- To quantize  $B_i$  we use quantization vector  $q$ . Each element  $q[n], n=0, 1, \dots, L_b-1$ , of  $q$  is a positive integer in a specified interval and represents the quantization step size for the coefficient  $B_i[n]$ . The elements  $\hat{B}_i[n]$  of the quantized DCT block  $\hat{B}_i$  obtained by the following operation:

$$\hat{B}_i[n] = B_i[n] // q[n], n=0, 1, \dots, L_b-1 \quad (17)$$

$$i=0, 1, \dots, N_b-1$$

where  $//$  represents division followed by rounding to the nearest integer.

- The lossless encoding of the quantized DCT coefficients involves run-length encoding, because the

quantization normally generates many null values followed by an entropy encoder.

The plot is shown in Fig. 5.

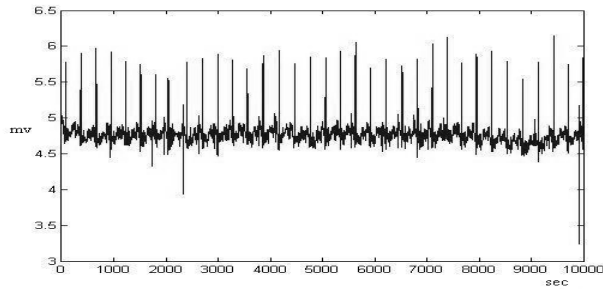


Figure 5(a). The ECG signal after DCT-II compression

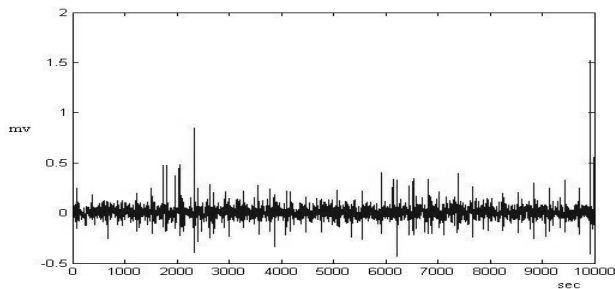


Figure 5(b). The Error signal after DCT-II compression

Figure 5. DCT-II compression analysis

## VI. results and discussion

We used data in the CSE database to test the performance of the four coding techniques. The ECG data is sampled at 333 Hz. The amount of compression is measured by CR and the distortion between the original and reconstructed signal is measured by PRD. The comparison table shown in Table 1, details the resultant compression techniques. This gives the choice to select the best suitable compression method. A data compression algorithm must represent the data with acceptable fidelity while achieving high CR. As the PRD indicates reconstruction fidelity; the increase in its value is actually undesirable. Although DCT-II provides maximum CR, but distortion is more. So a compromise is made between CR and PRD.

TABLE I. COMPARISON OF COMPRESSION TECHNIQUES

Method	Compression Ratio	PRD
DCT	80.8906	0.9346
FFT	89.5767	0.0123
DST	70.4073	1.1871
DCT-II	95.7700	1.3319

## VII. conclusions

Among the four techniques presented, DST provides lowest CR and distortion is also high. DCT improves CR and lowers PRD. Next is FFT which gives higher CR 89.5767 with PRD as 0.0123. But DCT-II provides an improvement in terms of CR of 95.77 but PRD increases up to 1.3319. Thus an improvement of a discrete cosine transform (DCT)-based method for electrocardiogram (ECG) compression is presented as DCT-II in terms of amount of compression. The appropriate use of a block based DCT-II associated to a uniform scalar dead zone quantiser and arithmetic coding show very good results, confirming that the proposed strategy exhibits competitive performances compared with the most popular compressors used for ECG compression.

## Acknowledgment

The authors would like to thank UGC, New Delhi for providing financial support to carry out this research work.

## references

- [1] Pranob K. Charles and Rajendra Prasad K. (2011): A Contemporary Approach For ECG Signal Compression Using Wavelet Transforms. *Signal and Image Processing: An International Journal (SIPIJ)*. Vol. 2, No. 1, 178-183.
- [2] Yucel Kocyigit, Mehmet Korurek and Bekir Karlik (1999): ECG Data Compression by Artificial Neural Networks. In *ELEC'99 International Conference On Electrical And Electronics Engineering*. E01.44/D-10, 338-339.
- [3] S. Jalaeddine, C. Hutchens, R. Stratan, and W. A. Coberly (1990): ECG data compression techniques-A unified approach. *IEEE Trans. Biomed. Eng.*, **37**, 329-343.
- [4] J. R. Cox, F. M. Nolle, H. A. Fozzard and G. C. Oliver (1968), AZTEC: A preprocessing scheme for real time ECG rhythm analysis. *IEEE Tran. Biomed. Eng.*, vol-BME-15, 128-129.
- [5] M. Sabarimalai Manikandan and S. Danpat (2006): Wavelet Threshold based ECG Compression using USZZQ and Huffman Coding of DSM. In *Science Direct Biomedical Signal Processing and Control*. 261-270.
- [6] B. R. S. Reddy and I. S. N. Murthy (1986): ECG data compression using Fourier descriptors, *IEEE Trans. Bio-med. Eng.*, BME-33, 428-433.
- [7] Mrs. S. O. Rajankar and Dr. S. N. Talbar (2010): An Optimized Transform for ECG Signal Compression. In *Proc. Of Int. Conf. on Advances in Computer Science*, 94-96.
- [8] J. Abenstein and W. Tompkins (1982): A new data-reduction algorithm for real time ECG analysis. *IEEE Tran. On Biomed. Engg.*, 29(BME-1):4, 3-8.
- [9] N. Ahmed, T. Natarajan and K. R. Rao (1974): Discrete Cosine Transform. *IEEE Trans. Trans. On Computers*. C-23, 90-93.
- [10] K. R. Rao and P. Yip (1990): Discrete cosine transform – algorithms, advantages, applications, San Diego: *Academic Press*.
- [11] M. Clausen and U. Baum (1993): Fast Fourier Transforms. *BI-Wiss.-Verl.*
- [12] L. Auslander, E. Feig and S. Winograd (1984): Abelian Semi-simple Algebras and Algorithms for the Discrete Fourier Transform. In *Advances in Applied Mathematics*.5, 31-55.
- [13] Tinku Acharya and Ajoy K. Roy. Image Processing Principles and Applications. *John Wiley*.

- [14] S. Chan and K. Ho (1990): Direct Methods for computing discrete sinusoidal transforms. *IEEE Proceedings*, 137, 433-442.
- [15] G. Steidl and M. Tasche (1991): A Polynomial approach to Fast algorithms for Discrete Fourier –cosine and Fourier-sine Transforms. In *Mathematics in Computation*, 56 (193), 281-296.
- [16] E. Feig and S. Winograd (1992): Fast Algorithms for Discrete Cosine Trnsforms. *IEEE Tran. On Signal Processing*.vol-40(9), pp 2174-2193.
- [17] Xuancheng Shao and Steven G. Johnson (May 10, 2007): Type-II/III DCT/DST algorithms with reduced number of arithmetic operations. *Preprint submitted to Elsevier*.