British Journal of Mathematics & Computer Science
4(10): 1365-1380, 2014

SCIENCEDOMAIN *international*
www.sciencedomain.org

Three Audio CODECs Using the LSM Interpolation and Comparison with PCM

Sameh Abdelwahab Nasr Eisa[1*]

[1]*Department of Mathematics, New Mexico Institute of Mining and Technology, 801 Leroy Place, PO #3696, Socorro, NM, 87801, USA.*

Original Research Article

Received: 24 August 2013
Accepted: 01 March 2014
Published: 24 March 2014

Abstract

In this paper we take advantage of the known interpolation Least Square Method (LSM) to construct audio coding/decoding (CODEC) algorithms. The purpose of this algorithm is to compress audio data, maintain high quality audio, and enable sending/storing audio as serial data through digital transmission systems. Our proposed algorithms can be an efficient replacement for the quantization process used in many CODECs and modulations like PCM. We clarify our research by explaining the reasons, the assumptions, and the experiments' results for each step individually. We applied the algorithms to 20 audio files and introduced three algorithms that approach the most efficient compression ratio in addition to best signal to noise ratio. We showed Pseudo Codes, Flowcharts, and complete results of some experiments, and also a comparison with PCM used in telephony system.

Keywords: Audio, Speech, Processing, CODEC, Interpolation, Polynomials, Modeling.

1 Introduction

Audio processing and coding is a very wide area of interest. Many researchers are trying continuously to improve existing audio CODECs, where others are trying to create new ones. Improving the quality as well as increasing the compression efficiency is always a target. Recently, many attempts have been made in the processing world using interpolation methods.

Yung-Gi Wu, Chia-Hao Wu in [17] tried Lagrange interpolation in image processing. Shugang Wei in [14] presented a signal level compressor using Newton's interpolation formula. McPherson, T,Jr. in [12] presented compression in PCM using interpolation. Additionally, Bhaskar, U in [1] worked with interpolation in his processes. We have cited a number of published works that studied MP3 and MPEG systems and some successful attempts published lately to keep the achievements made by other researchers. Our algorithms for CODECs are compared with PCM, because PCM uses the quantization process which we suggest replacing it with our process.

**Corresponding author: seisa@nmt.edu, sameheisa235@hotmail.com;*

We propose three approaches for audio CODECs that can be applied to storing/transmitting audio data. We use the known Least Squares Method (LSM) interpolation as a bases for these approaches. The three CODECs also could be performed with/without some of the other known audio CODECs such as, MP3. Our algorithms are applied to audio sample values, so they can be added as a part in the coding/decoding, as we tried them on MP3 audios. Efficient compression ratio, besides high quality audio after recovery, were experimented on 20 audio files that presented human voices with/without quiet music and songs with different types of music. We made specific comparison with PCM used in telephony system because we think our algorithm can replace the quantization part found in some CODECs/modulations like PCM. The comparison has shown progress achieved by our work.

We prefer to show in this paper explicitly the steps we made from the original assumptions to the final algorithms to clarify our methodology during the research procedures. We clarified these steps by discussing the reasons, expectations, and showing the results through experimented files.

2. Fundamental Basics and Assumptions

We show clearly the mathematical basis used, in addition to assumptions we have considered in our work.

2.1 Modeling the Values of Audio Samples

The samples' values form the behavior and the pattern of the audio, and they are defined by specific value per sample. We propose models for these data values using the interpolation LSM. After some experiments with polynomials, exponential, and power models (discussed in section 3) we selected the polynomial models to represent the audio data because LSM with polynomial models have shown the best fitting for samples' values.

$$y_i \cong a_0 + a_1 x_i + a_2 {x_i}^2 \dots a_n {x_i}^n \tag{1}$$

Where y is a vector (row of data) that contains audio's sample values, x is a vector that contains the values in which is presented over it, and will be considered (number of samples ordered 1, 2, 3.…N_s), n is the polynomial degree and i is the index ranged from 1-N_s.

We usually know the values of $x\ and\ y$ by considering polynomial models. We will have to find $a_0, a_1, a_2 \dots a_n$ by solving a system of linear equations, which can be represented in matrix form

$$Y = XA \tag{2}$$

$$A = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}_{(n+1)x1} , \; Y = \begin{bmatrix} \sum_{i=1}^{N_s} y_i \\ \sum_{i=1}^{N_s} x_i y_i \\ \sum_{i=1}^{N_s} {x_i}^2 y_i \\ \vdots \\ \sum_{i=1}^{N_s} {x_i}^n y_i \end{bmatrix}_{(n+1)x1}$$

$$\& \; X = \begin{bmatrix} N_s & \sum_{i=1}^{N_s} x_i & \sum_{i=1}^{N_s} {x_i}^2 & \cdots & \sum_{i=1}^{N_s} {x_i}^n \\ \sum_{i=1}^{N_s} x_i & \sum_{i=1}^{N_s} {x_i}^2 & \sum_{i=1}^{N_s} {x_i}^3 & \cdots & \sum_{i=1}^{N_s} {x_i}^{n+1} \\ \sum_{i=1}^{N_s} {x_i}^2 & \sum_{i=1}^{N_s} {x_i}^3 & \sum_{i=1}^{N_s} {x_i}^4 & \cdots & \sum_{i=1}^{N_s} {x_i}^{n+2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sum_{i=1}^{N_s} {x_i}^n & \sum_{i=1}^{N_s} {x_i}^{n+1} & \sum_{i=1}^{N_s} {x_i}^{n+2} & \cdots & \sum_{i=1}^{N_s} {x_i}^{2n} \end{bmatrix}_{(n+1)x(n+1)} \quad (3)$$

Then

$$A = X^{-1}Y \quad (4)$$

Therefore, we deduce the coefficients of the polynomials using Eq. (4) by applying any numerical scheme to solve the system.

In order to validate the model, we have to determine the value r (coefficient of determination) which clarifies the fitting of the proposed polynomial corresponding to the real data values (audio samples values).

$$r = \sqrt{\frac{\sum_{i=1}^{N_s} (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{N_s} (y_i - \bar{y})^2}} \quad (5)$$

Where $\hat{y}_i$ is the theoretical y-value corresponding to x_i (calculated through the polynomial) and $\bar{y}$ is the mean value of all experimental y-values. Whenever $r$percentage approaches 100%, then we have perfect fitting with no error at all. This value can be used to identify whether the proposed polynomial presents the audio data efficiently or not. We will suggest various ways in section 3 to apply this modeling on audios.

2.2 Basic Assumptions

Our method is concentrated on reducing the length of the vector y, which contains the audio's sample values. In order to do that, we will apply LSM with several aspects as a base for different algorithms. We store/transmit the values of the polynomial coefficients instead of the vector of the audio's sample values. All of these algorithms have the following specific assumptions in common.

1. During the encoding, the algorithms are repeatedly applied to a set of N_s samples. The number of samples N_s can be the same or changing every time. The algorithms applied several times until all the samples are processed.
2. The compression ratio "CMR" will be calculated as
   $CMR = length(\text{vector of coefficients})/length(\text{vector of audio}' \text{ samples})$
3. The values of the coefficients must be in the numeric range of the audio' sample values. For example, if we scaled the samples' values [-1, 1] the coefficients must be in this

range. The differences between them could be represented by known number of levels like those in the audio's values themselves. We assume this constraint to ensure the ability to use our algorithms for digital storing/transmission.

4. The method could be applied with/without some of the other CODEC techniques because our method depends on processing the samples' values. It could be applied directly after a voice sampling process or after coding the audio values with some other techniques, to reduce the audio storage vector.
5. Durations of silence are one of the known speech characteristics. Taking this in to account, would make some of our algorithms more efficient because we have many zero samples.
6. To ensure compression ratio, we have to restrict our choice for N_s to follow the inequality $N_s \geq n+1$ in each operation, where n is the polynomial degree.
7. Mostly, there is no difference between using different n with the same CMR in the algorithms; we will work with $n = 4$. However, using other assumption for n will not damage the results or affect the quality negatively. We recommend that the choice of n follows the inequality $2 \leq n \leq 10$ to ensure a better adaptive model, as well as to be far from oscillating degrees of high n (investigated insection 3).

3. Steps of the Research Including Discussions and Experiments

Our research went through several steps; some of them to confirm the basic assumptions (section 2.2), some to identify specific features for operations, and also steps to test possible algorithms deduced from using LSM. Twenty audio files were tested through the steps, 11 files for human voices and 9 music files, listed in Table.1

Table 1. Experimented files

File Index	Description	Duration (sec)	n-bits/sample	$F_s(Hz)$
M.1	Loud Music	5	16	44100
M.2	Quiet Music	5	16	44100
M.3	Quiet Music + Arabic male voice	6	16	44100
M.4	Old Music + Arabic male voice	5	16	44100
M.5	Old Music + English male voice (Dean Martin)	5	16	44100
M.6	Music + English female voice (Celine Dion)	5	16	44100
M.7	Loud Music + English female voice (Shakira)	5	16	44100
M.8	Old Music + English male voice (Louis Armstrong)	5	16	44100
M.9	Audience cheering + indistinct music	5	16	44100
H.1	Male voice (Morgan Freeman)	30	16	44100
H.2	Female voice (Anne Hathaway) + Male voice (Patrick Warburton)	8	16	44100
H.3	Male voice (Robert Di Nero) + Loud noise	5	16	44100
H.4	Female voice (Susan Sarandon)	14	16	44100
H.5	Female voice (Susan Sarandon) + Quiet music	41	16	44100
H.6	Female voice (Suzanne Pleshette)	6	16	44100

H.7	Male voice (Heath Ledger)	11	16	44100
H.8	Male voice (Heath Ledger)	31	16	44100
H.9	Female voice (Jennifer Garner) + Male voice (Ricky Gervais)	11	16	44100
H.10	Female voice(Kate Ashfield) + male voice(Simon Pegg)	11	16	44100
H.11	Male voice(Michael Caine) + Male voice (Christian Bale)	21	16	44100

These files are experimented using the Matlab program in which sample values are scaled from [-1, 1]. We explain, discuss, and show the results of every step.

Step1. Random experiments to consider the model type and the parameters

By scaling random distribution of values between [-1, 1] that resemble the audio values, we conducted several experiments to determine the best modeling type (polynomials, exponential, or power models), which would fit the audio data by satisfying higher r Eq. (5). Also, for every type of modeling we could change the distribution of x values for the same number of samples, to test changes in CMR.

Notes and details of the experiments in step 1;
1. By taking the same distribution of $x$multiple times with different values of y each time, it was clear that the polynomials model achieved highest r. Table 2 presents some results of experiments made on M.1, by calculating average values of$r$ along the audio file. We made that by applying LSM with different models for $N_s = 10$ for every operation until all samples are processed. 2. By changing the distribution of x to be in linear/non-linear sequences different from the usual sequence (1,2,3…N_s), we did not observeany negative/positive effects.
Results and conclusions;
1. We will consider polynomials as a basic LSM for the following steps. 2. Changing the x distribution will not affect the results, so it will be kept as linear distribution.

Table 2. Results of experiment for testing types on the file M.1

Model	N_s	r	Model	N_s	r
Exponential	10	0.69099	Poly. 4nd	10	0.96524
Linear	10	0.69168	Poly. 5nd	10	0.98087
Poly. 2nd	10	0.89188	Power	10	0.65869

Step 2. Choice of polynomial degree

It is obvious from the previous step that, polynomial types give more efficient fitting. In Table 2 we assumed $N_s = 10$. It is expected that, when polynomial degree increases with the same number of points, the accuracy of interpolation is increasing too due to known interpolation characteristics. Our research concentrated on modeling the audio data to balance between CMR and high quality. So we then compared different polynomial degrees with the same CMR to test the quality. In Table 3. We considered $CMR = 1/3$ and we observed the error value to see which n is to be considered.

Notes and details of the experiments in step 2;
1. By considering $CMR = 1/3$ we consider $\frac{n+1}{N_s} = \frac{1}{3}$, where n is the polynomial degree and N_s is the samples. 2. We apply the experiments to the same files and try n from 1-15 and N_s will be (6, 9, 12, 15….48) respectively. One of these experiments on H.1 is presented in Table.3. 3. The experiments indicated that n for the same CMR will not make any differences. 4. The accuracy decreased after $n = 10$ most likely due to oscillation of higher degrees in interpolation.
Results and conclusions;
1. The experiment confirmed basic assumption 7 in section 2.2 regarding the choice of n. 2. We will consider $n = 4$ in the experiments in the next steps.

Table 3. Experiments on H.1 related to step2

Poly, degree	N_s	Average diff	Poly. degree	N_s	Average diff
1	6	0053935	9	30	0052454
2	9	0051803	10	33	0052008
3	12	0050845	11	36	0052566
4	15	005148	12	39	0055419
5	18	0051581	13	42	0059939
6	21	0052247	14	45	0067355
8	27	005266	15	48	0079257

Step 3. Applying LSM with $n = 4$ and fixed N_s

After we considered LSM with polynomials model, we worked with $n = 4$. We applied LSM on fixed N_s until we finished the whole file. We changed N_s several times to have different CMR as we calculated $CMR = \frac{n+1}{N_s}$. We tried $N_s = 10,15,20,25,30$ which mean that we have $CMR = \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{5}, \frac{1}{6}$ respectively. We calculated average values for r for each CMR tested. Then, if we found interesting results, we made encoding to the audio vector and stored the resultant coefficients in another vector. After that, we decoded the coefficients to deduce recovered values of the original audio. We had to calculate SNR and hear the difference in quality between the original and the recovered files.

Notes and details of the experiments in step 3;
1. We started with $CMR = \frac{1}{2}$ in all files and we achieved high fitting by calculating the average value of r in each file as shown in Fig. 1. 2. We also tried $CMR = \frac{1}{3}, \frac{1}{4}, \frac{1}{5}, \frac{1}{6}$ and it was clear that $CMR < \frac{1}{4}$ gives non-accurate fitting as average value of r decreased. Fig. 1 show average values of r for $CMR = \frac{1}{3}\ and \frac{1}{4}$ respectively. 3. We designed an algorithm for encoding and decoding with $n = 4\ and\ N_s = 10,15,20$ and observed SNR and heard the recovered audio file. Maximum SNR for $CMR = \frac{1}{2}, \frac{1}{3}, \frac{1}{4}$ respectively are presented in Fig. 1. 4. No audible quality difference could be discerned for $CMR = \frac{1}{2}$. As it is clear also in the SNR values which were high as shown in Fig. 1.

5. The sound and the music were recognized in the case of $CMR = \frac{1}{3}, \frac{1}{4}$, but there were quality differences we could sense. These differences were inversely proportional with CMR. However, in speech files it was hard to sense differences in $CMR = 1/3$, unlike the music files, but these differences are not critical, even though SNR were measured high as shown in Fig. 1.
Results and conclusions;
1. We will consider the CODEC in this step as our first algorithm (detailed in 4.1). 2. We can consider the CODEC in this step to give very high quality with $CMR \geq \frac{1}{2}$.

Step 4. Applying LSM with $n = 4$ and non-fixed N_s to ensure the quality

The algorithm we considered in the previous step would apply LSM on a fixed number of $N_s$each operation. We thought that after taking $n = 4$, we might have two different cases. First case, a long sequence of audio samples may have small rate of change. In this case, using higher $N_s \gg n + 1$ maintains quality and high CMR. For example, silence duration produces long sequence of zeros, or constant tone rates. Second case, some sequences with high oscillating values require small N_s to maintain quality, but in this case The CMR will be low. For example, loud noise or screaming.

Notes and details of the experiments in step 4;
1. We specified "$diff$" which will be the difference not to beexceeded between the original values and the recovered ones. 2. We developed our code to scan maximum number of $N_s$that gives error $\leq diff$. 3. The definition of CMR will change during this step. Because we will need two vectors, one for storing the coefficients and one for storing N_s we used every operation. As a result, $CMR = \frac{length(coefficients\ vector)+length(N_s\ vector)}{length(original\ audio\ vector)}$. 4. In order to know if we can consider this modification as another algorithm and CODEC method, we have to reach CMR like the first algorithm and average value of r must be in the same range. 5. We changed the value of $diff$ until we reached the same CMR like the first algorithm and then we measured the quality through experimenting with some files. The results are presented in Fig. 1. 6. The quality enhancement was audible.
Results and conclusions;
1. We will consider another algorithm and second CODEC method from this step (detailed in section 4.2). 2. This CODEC technique can be applied to ensure the quality for applications that have a preferred specific amount of tolerance in the error.

Step 5. Applying LSM after changing the distribution of the sample values by summing them

The accuracy of the interpolation depends mainly on the distribution of the sample values. We suggested reaching convergence between sample values by reconstructing the audio vector, and replacing each sample value with the sum of this sample and the previous one in the vector. By applying this procedure, then applying the first algorithm we expected to improve the interpolation of the new reconstructed vector compared with the original vector.

Notes and details of the experiments in step 5;
1. We consider $CMR = \frac{1}{2}$ and $N_s = 10$ like the first algorithm. 2. We applied the new procedure and deduce average value of r which has shown significant improvement compared with the previous algorithms (Fig. 1). 3. We designed an algorithm for encoding and decoding, then we measured SNR and quality of the recovered data as shown in Fig. 1, which has shown significant improvement and clear enhancement in hearing.
Results and conclusions;
1. We will consider this algorithm as our third CODEC from this step (detailed in section 4.3). 2. This CODEC can be applied to ensure the quality and to improve the first algorithm's results by involving more processing.

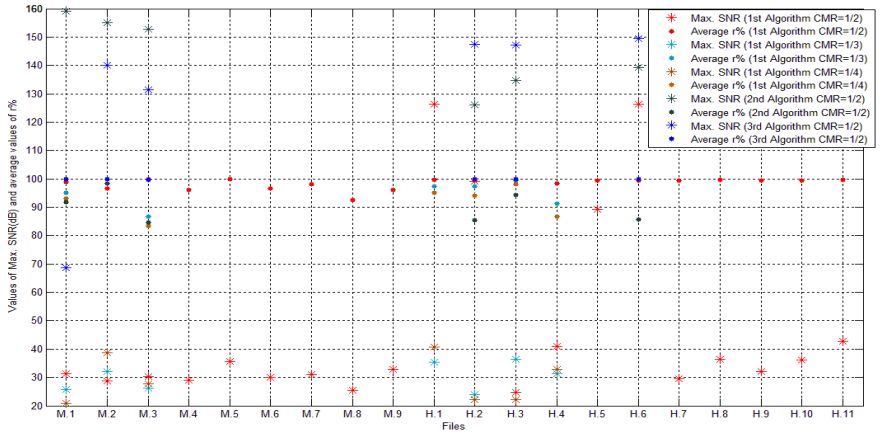

Fig. 1. Collective maximum SNR and average r% for the results from the three algorithms (steps 3,4,5)

Step 6. Applying LSM toMP3 files

In this step we tried the algorithm step 3 with $CMR = \frac{1}{2}$ to MP3 files. This was to confirm the basic assumption 4 by testing MP3 files.

Notes and details of the experiments in step 6;
1. We experimented several files, which were not from the files in Table 1., because we preferred full length songs to ensure the ability of our algorithm to be applied to MP3 files. We measured average values of r and SNR, which are shown in Fig. 2.
Results and conclusions;
1. We can apply our algorithm in phase with/without other CODECs, if they generate value vectors for the audio.

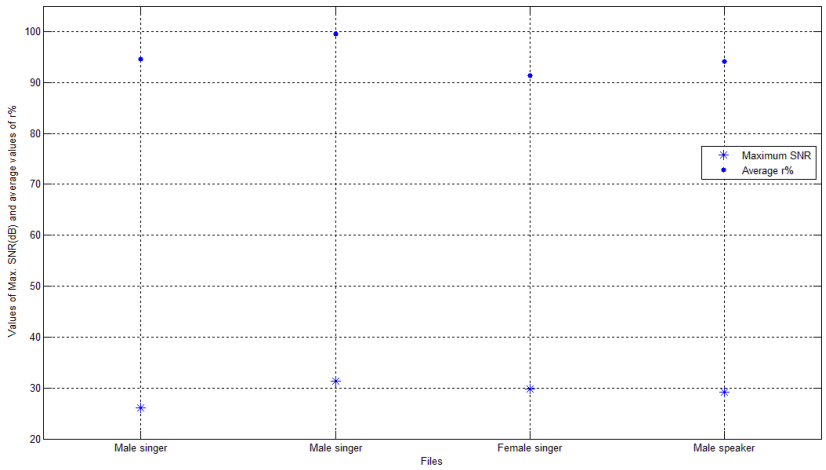

Fig. 2. Average r% and maximum SNR for tested MP3 files

4. The Three Algorithms Considered as "CODEC"

We considered three algorithms to make encoding/decoding for the audio data. We will clarify them using pseudo codes and flowcharts (Figs. 3, 4, 5)**.**

a. The first algorithm (results from step3)

Encoding part;	Decoding part;
Input: y, n, N_s Step1: Create empty array for compressed values (store the coefficients). Step2: For each N_s sample in y**:** a. Perform LSM with n degree on picked samples. b. Concatenate the resulting coefficients with compressed. Step3: Return compressed.	Input: $compressed, n, N_s$ Step1: Create empty array for y (recovered values). Step2: For each $n+1$ elements in $compressed$ (the coefficients): a. Deduce N_s elements present sample values by evaluating n polynomial. b. Concatenate the resulting sample values with y. Step3: Return y.

b. The second algorithm (results from step4)

Encoding part;	Decoding part;
Input: $y, n, diff$. Step1: Create two empty arrays, one for compressed and one for samples. Step2: Search for maximum number of samples in y after performing LSM and recover ymax $(differences) < diff$. Step3: Store the coefficients in $compressed$ and number of samples used in $samples$. Step4: Repeat step2 and step3 for the whole file. Step5: Return $compressed$ & $samples$.	Input: $compressed, samples, n$ Step1: Create empty array for y (recovered values). Step2: For each element value (N) in $samples$**.** a. Evaluate N samples values using $n+1$ elements from $compressed$ b. Concatenate the results with y. Step3: Store the coefficients in $compressed$ and number of samples used in $samples$. Step4: Return y.

c. The third algorithm (results from step4)

Encoding part;	Decoding part;
Input: $, n, N_s$. Step1: Create two empty arrays, one for compressed values (store the coefficients). Step2: For each N_s samples in y**:** a. For each element in the N_s. i. $sample(i) = sample(i) + sample(i-1)$. b. Perform LSM with n degree on the modified samples. c. Concatenate the resulting coefficients with compressed. Step3: Return $compressed$.	Input: $compressed, n, N_s$ Step1: Create empty array for y (recovered values). Step2: For each $n+1$ elements in $compressed$ (the coefficients): a. Deduce N_s elements present sample values by evaluating n polynomial. b. For each sample in N_s (in reverse order) i. $sample(i) = sample(i) - sample(i-1)$. c. Concatenate the resulting samples values with y Step3: Return y.

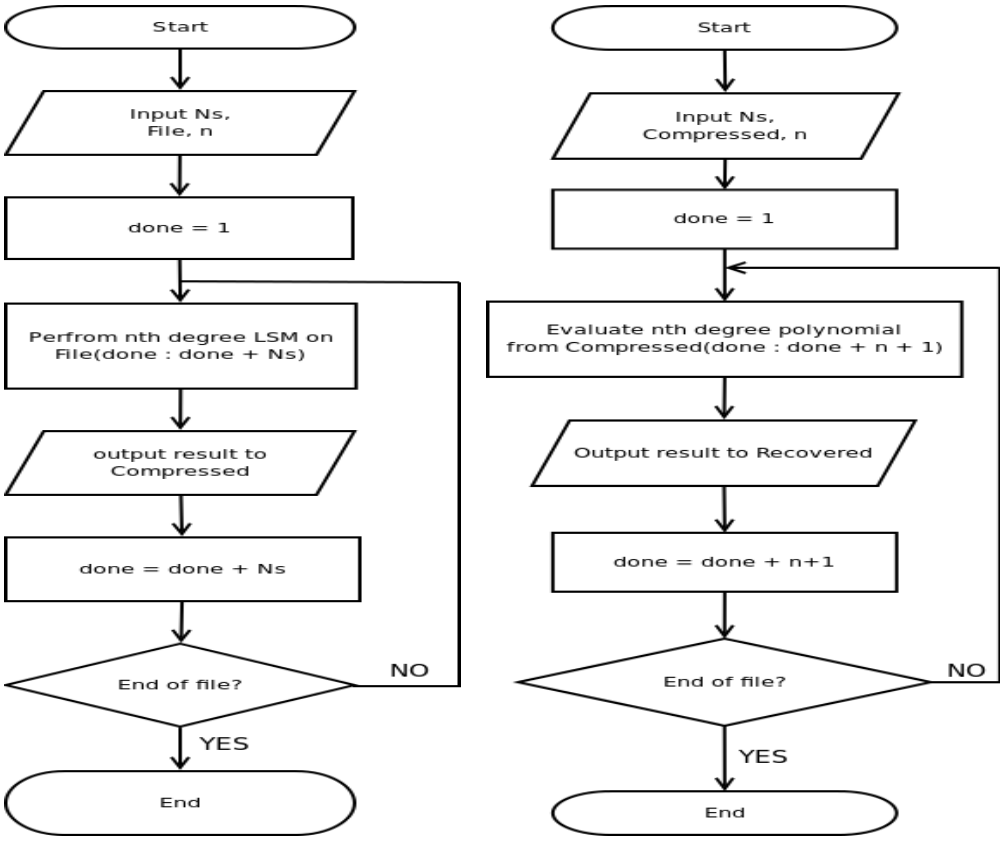

Fig. 3. Flow-chart for the code of the 1st algorithm encoding & decoding from left to right.

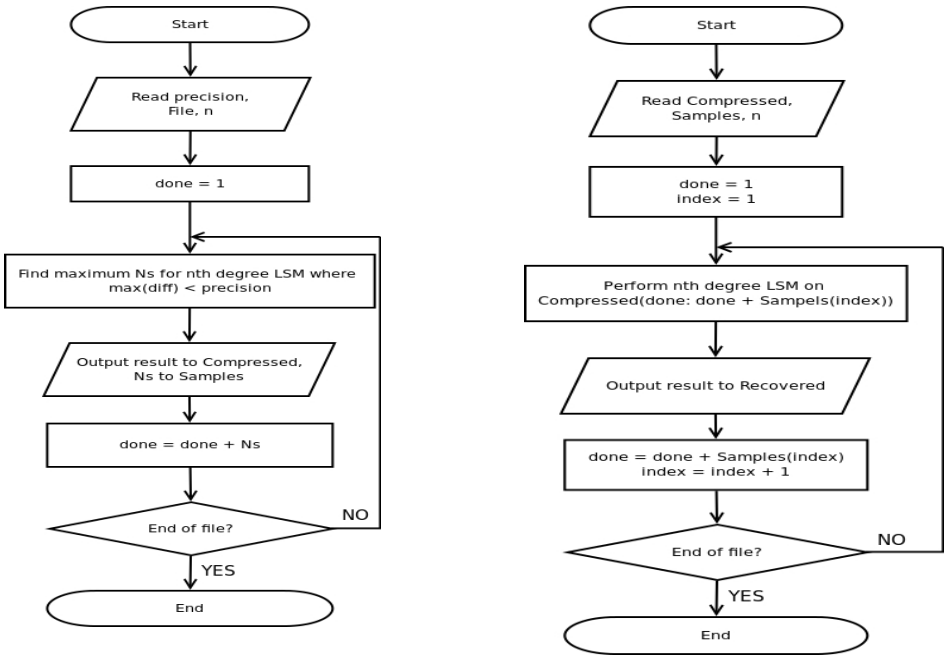

Fig. 4. Flow-chart for the code of the 2nd algorithm encoding & decoding from left to right.

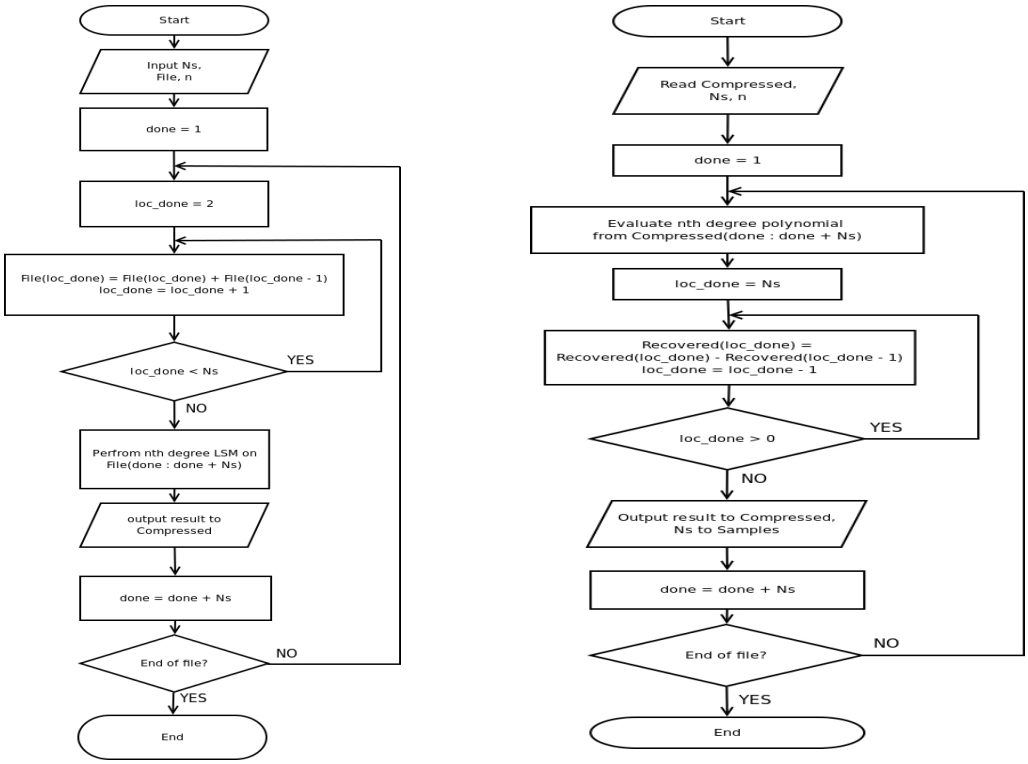

Fig. 5. Flow-chart for the code of the 2nd algorithm encoding & decoding from left to right.

5. Results and Comparison with PCM

In this section, we present some results obtained by using the algorithms constructed in this research. We will make comparisons between our algorithm and PCM used in telephony system.

5.1 Results

We present in Fig. 6 the results of applying the first algorithm using $n = 4$ and $N_s = 10$ ($CMR = 0.5$) on three audio files. The recovered audio resembles the original one in the quality and we didn't detectan audible difference. It is clear that SNRs are high.

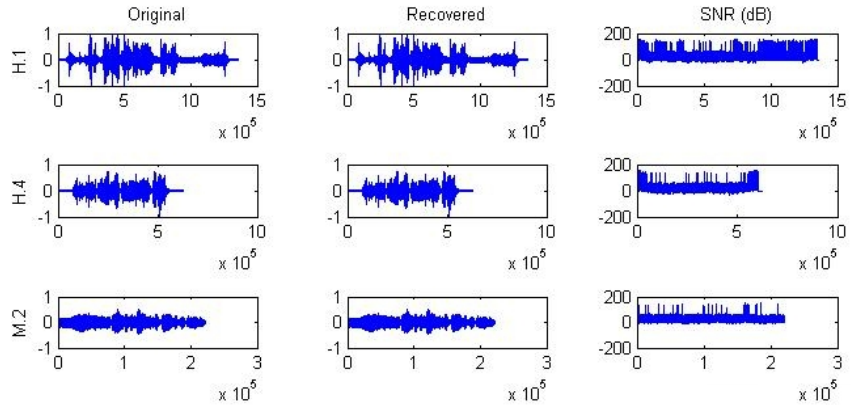

Fig. 6. Full experiments' results using our first algorithm on three audio files

We present in Fig. 7 the results of applying the first algorithm by using $n = 4$ and $N_s = 15,20$ and $CMR = \frac{1}{3}, \frac{1}{4}$ on H.1. We sense some differences in the quality, however, the speech was recognizable and SNR is relatively high.

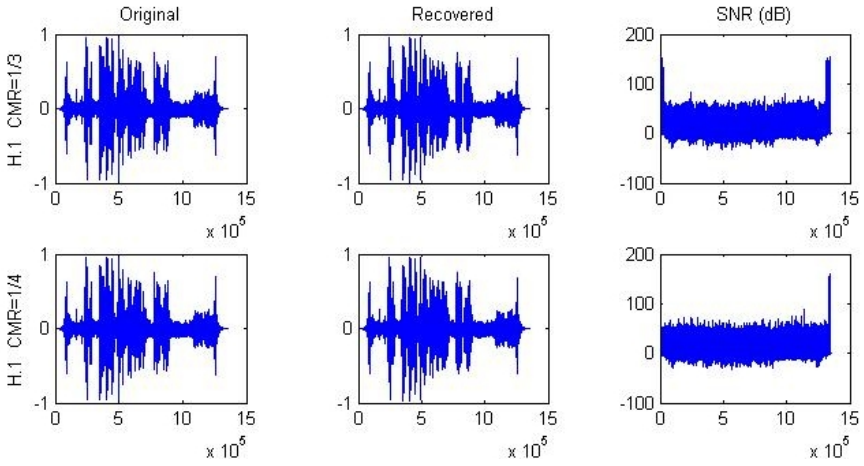

Fig. 7. Full experiments' result using first algorithm on file with different CMRs

We present in Fig. 8 the recovered audio Using the first and second algorithms with $CMR = \frac{1}{3}$. It is clear that the SNR in the second algorithm shows more consistency because we designed this algorithm for obtaining high and consistent qualityacross the samples.

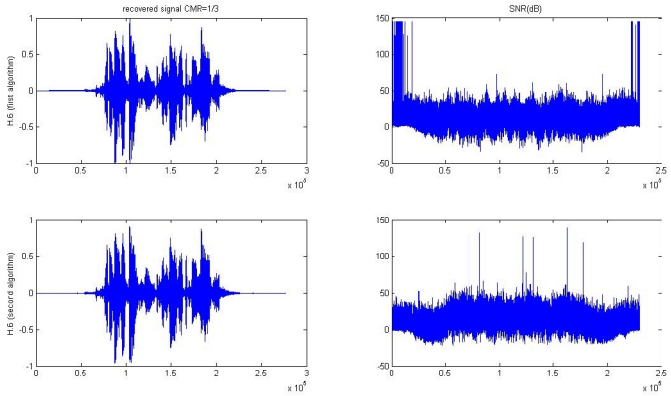

Fig. 8. Comparison between first and second algorithms on the same file

5.2 Comparison with PCM

The telephony system uses PCM concept as a transmission modulation technique. We preferred to compare our simplest CODEC, which is the first algorithm, with the results obtained using PCM. We believe that the ability of adding our algorithm directly to the sample values after sampling process will be better than applying the quantization process used generally in many CODECs and specifically in PCM. Consequently, we applied PCM to several of our files and in our files' case that will produce $CMR = 0.5$ because bit rate was 16 and in telephony system it's usually is 8. We applied our first algorithm to obtain $CMR = 0.5$ on the same files. We achieved higher average SNR in most files and significantly higher maximum SNR in all files. We also clarified the difference between our algorithm and PCM by comparing specified amount of samples to show our superior results. Fig. 9 shows the comparisons of average and maximum SNR.

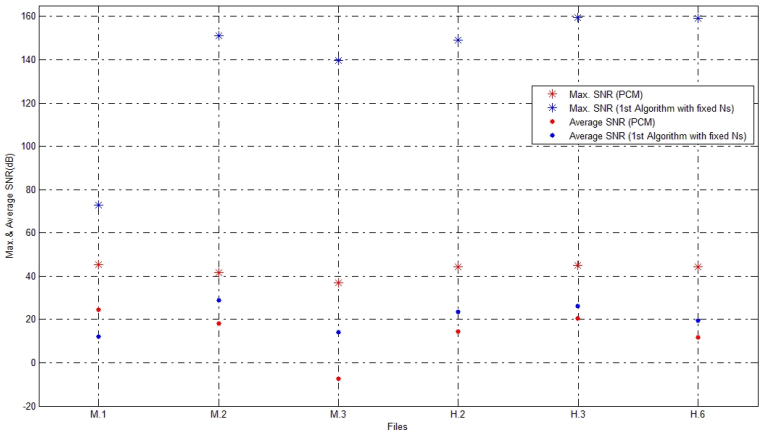

Fig. 9. Comparisons between our algorithm and PCM on the same files

We applied our first algorithm and PCM to 100 samples from one file, and compared the recovered sample values, and showed SNR across the samples. These comparisons are presented in Fig. 10.

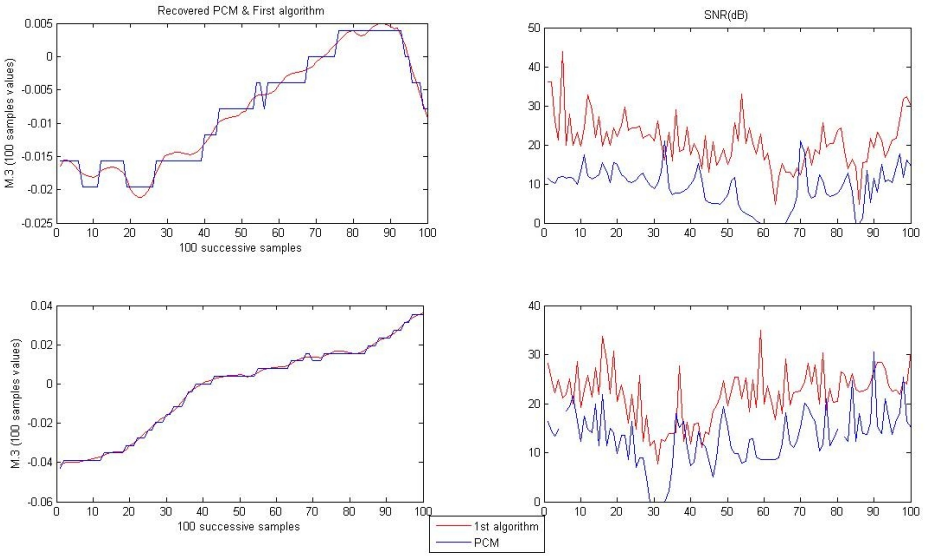

Fig. 10. Comparisons of our first algorithm with PCM over 100 samples from audio file

6. Conclusion and Expectations

We introduced three algorithms for encoding/decoding audio data (CODEC). We used the interpolation method known as LSM to deduce these algorithms. The first algorithm is based on applied LSM on fixedN_s (number of samples) until we finished the data and stored values (coefficients) each time. The second algorithm improved the first one by initializing a specific value of maximum absolute difference allowed between the original and the recovered values to ensure the quality across the data, taking advantage from some speech characteristics like silence durations. The third algorithm works like the first one but in a different distribution of sample values. The new distribution results from adding each sample's value to the previous one to generate a new vector to increase convergence between values of the new vector. Pseudo-codes and flowcharts are proposed in this research for every algorithm. We presented average value of r Eq. (5) and SNR for many files (including MP3 files) tested by the three algorithms. Results from experiments that clarify and prove some basic assumptions we considered were presented.

Clear comparison between our simplest algorithm (the first) and PCM used in telephony system have shown progress of our work. The values of the coefficients stored resemble the values of the audio's samples in scale and sensitivity. This indicates that the values of the coefficients can be used not only in compression applications but also in transmission applications. They can be sent as serial data with the same digital sequences known in quantization levels, but with higher quality (same compression), as we presented in this research. We believe that our algorithms can replace the quantization process in any CODEC depending on the quantization process.

The methodology we used in this research is the basic for further research. We are conducting in image and video compression. We expect that this path will open a lot of applications and ideas that depend on LSM.

Acknowledgements

The author would like to show gratefulness to Waleed Lotfy, bachelor student in Faculty of Engineering, Alexandria University, Egypt for his enormous contribution in computer work (coding, testing) and in documenting through this research.

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Bhaskar U, Low bit-rate voice compression based on frequency domain interpolative techniques, Audio, speech, and language processing, IEEE Transactions. 2006;2:558-576.

[2] Cheng-Yu Yeh, Chang-Zhi Zhuo. An efficient complexity reduction for G.729 speech codec, Computers and Mathematics with Applications. 2012;64:878-896.

[3] Demetrios Okkalides. Assessment of commercial compression algorithms, of the lossy DCT and lossless types, applied to diagnostic digital image files, Computerized Medical Imaging and Graphics. 1998;22:25-30.

[4] Din-Yuen Chan, Cheng-Yuan Ku, A low-complexity, high-quality, 64-Kbps audio codec with efficient bit allocation, Digital Signal Processing. 2003;13:23–41.

[5] Dongge Li, Ishwar K. Sethi, Nevenka Dimitrova, Tom McGee. Classification of general audio data for content-based retrieval, Pattern recognition Letters. 2001;22:533-544.

[6] Eric D. Scheirer, Youngjik Lee, Jae-Woo Yang. Synthetic and SNHC audio in MPEG-4, Signal Processing: Image Communication. 2000;15:445-461.

[7] Ge Gao, Ching PC, Tan Lee. A new approach to generating pitch cycle waveform (PCW) for waveform interpolation codec, Microprocessors and Microsystems. 2002;25:421-426.

[8] Jeroen Breebaart, Gerard Hotho, Jeroen Koppens, Erik Schuijers, Werner Oomen, Steven Van De Par, Background, Concept, and Architecture for the Recent MPEG Surround Standard on Multichannel Audio Compression, J. Audio Eng. Soc. 2007;55(5).

[9] Karlheinz Brandenburg, Oliver Kunz, Akihiko Sugiyama, MPEG-4 natural audio coding, Signal Processing: Image Communication. 2000;15:423-444.

[10] Kris Hermus, Werner Verhelst, Philippe Lemmerling, Patrick Wambacq, Sabine Van Huffel. Perceptual audio modeling with exponentially damped sinusoids, Signal Processing. 2005;85:163–176.

[11] Contina L, Edler B, Meares D, Schreiner P. Tests on MPEG-4 audio codec proposals, Signal Processing: Image Communicarion. 1997;9:327-342.

[12] McPherson T, Jr. PCM speech compression via ADPCM/TASI, Acoustics, speech, and signal processing, IEEE international conference on ICASSP. 1977;77:184-187.

[13] Shijun Xiang, Hyoung Joong Kim, Jiwu Huang. Audio watermarking robust against time-scale modification and MP3 compression, Signal Processing. 2008;88:2372–2387.

[14] Shugang Wei. Audio dynamic range compression characteristics based on an interpolating polynomial, Circuits and systems, 2004, NEWCAS 2004, The 2nd Annual IEEE Northeast Workshop. 2004;133–136.

[15] Molla S, Torrésani B. A hybrid scheme for encoding audio signal using hidden Markov models of waveforms, Appl. Comput. Harmon. Anal. 2005;18:137–166.

[16] Vladimir Britanak. A survey of efficient MDCT implementations in MP3 audio coding standard: Retrospective and state-of-the-art, Signal Processing. 2011;91:624–672.

[17] Yung-Gi Wu, Chia-Hao Wu, Image vector quantization codec indices recovery using Lagrange interpolation, Image and Vision Computing. 2008;26:1171–1177.

[18] Zoran Peric, Jelena Nikolic. An adaptive waveform coding algorithm and its application in speech coding, Digital Signal Processing. 2012;22:199–209.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
www.sciencedomain.org/review-history.php?iid=466&id=6&aid=4090