Buletinul Ştiinţific al Universităţii "Politehnica" din Timişoara

Seria ELECTRONICĂ şi TELECOMUNICAŢII
TRANSACTIONS on ELECTRONICS and COMMUNICATIONS

Tom 49(63), Fascicola 2, 2004

Adaptive Filtering Algorithms

Erwin Szopos[1], Norbert Toma, Marina Ţopa

Abstract *– The paper presents some efficient algorithms for adaptive filtering: Wiener filter, Least Mean Square (LMS) algorithm, Kalman algorithm. These are used in several applications such as: echo canceller on telephone lines, enclosure noise canceller, adaptive equalization etc.*

The above-mentioned algorithms were implemented using the virtual instrumentation program LabVIEW, and Simulink, respectively. Experiments were carried out and their utility, limits and efficiency are demonstrated on different types of signals (Sine wave, audio signals).

Keywords: *Wiener filters, mean square error, performance area, steepest descent, convergence, Kalman filter.*

1. INTRODUCTION

Adaptive filtering is used when is necessary to realize, simulate or model a system which characteristics develop with time. This leads to the use of time variable coefficients filters. The variations of the coefficients are defined by an optimization criterion and are realized according to an adaptive algorithm. In the literature there are many different criteria and algorithms. The simplest but the most important in practice is the case where the criterion of mean square error minimization is associated with the gradient algorithm.

While the filtering with constant coefficients is generally associated with frequency domain specifications, the adaptive filtering corresponds to time domain specifications and is obvious to use it for the filter coefficient computation [2].

2. PRESENTATION OF THE ADAPTIVE ALGORITHMS

2.1 *The Wiener algorithm*

The principle of an *adaptive filter* consists in time variation and auto fitting of its characteristics. Usually, an adaptive filter takes the shape of a FIR filter structure, with an adaptive algorithm permanently updating the filter coefficients, when the *error signal* is minimized in accordance with a criterion.

The *Wiener* filter structure is shown in figure 1.

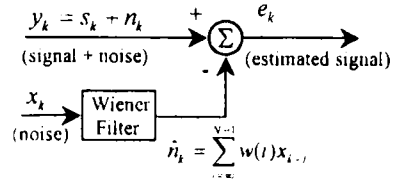

Fig. 1. *Wiener* filter structure

At the k moment, the sample y_k contains two components: the main signal s_k and a noise component n_k, which is correlated with x_k. The Wiener filter produces an optimum estimation of n_k, named $\hat{n}_k$. It is presumed that the *Wiener* filter is a FIR filter with N coefficients and the estimated error signal e_k is computed by subtraction of noise estimation $\hat{n}_k$ from the input signal y_k:

$$e_k = y_k - \hat{n}_k = y_k - \sum_{i=0}^{N-1} w(i) \cdot x(k-i)\,, \qquad (1)$$

where $w(i)$ are the Wiener filter coefficients. Because it operates with discrete values, the input signal and the filter coefficients can be represented in matrix form:

$$\begin{aligned} X_k &= \left[x_k \quad x_{k-1} \quad \ldots \quad x_{k-(N-1)}\right]^T \\ W &= \left[w(0) \quad w(1) \quad \ldots \quad w(N-1)\right]^T \end{aligned} \qquad (2)$$

By substituting these matrices in equation (1), the estimated error signal will be:

$$e_k = y_k - W^T X_k = y_k - X_k^T W\,. \qquad (3)$$

The instantaneous *biquadratic error* of the signal can be obtained by squaring equation (3):

$$e_k^2 = y_k^2 - 2W^T (y_k X_k) + W^T X_k X_k^T W\,. \qquad (4)$$

The *square mean error* (*SME*) ξ is defined by the probabilistic operator of the quadratic error from equation (4). Thus *SME* can be described:

[1] Technical University of Cluj-Napoca, Str. C-tin. Daicoviciu 15, 400020 Cluj-Napoca, Romania, Phone-fax: +40 264 591340, E-mail: erwin@bel.utcluj.ro

$$\xi = E\left[e_i^2\right] = E\left[y_i^2\right] - 2W^T E\left[y_i X_i\right] + W^T E\left[X_i X_i^T\right] W \quad (5)$$

The *SME* function can be expressed more suitable by substituting the term $E[X_k X_k^T]$ from (5) with the *autocorrelation matrix* R_{XX}. More, the term $E[y_k X_k]$ can be substituted with the *intercorrelation matrix* R_{yX}. Thus, the *SME* can be expressed as:

$$\xi = E\left\{e_k^2\right\} = E\left\{y_k^2\right\} - 2W^T R_{yX} + W^T R_{XX} W \quad (6)$$

From (6) we can observe that ξ is a *quadratic function* of the weights of vector W (filters coefficients). When equation (6) is expanded, the elements of W will be only of first and second order. This equation is valid when the input components and the desired response are stochastic (random) variables.

2.1.1 *Performance area*

A part of a bidimensional mean error function is shown in figure 2. The vertical axis represents the *SME* and the other two horizontal axes represent the values of two coefficients of the filter. The square error or the *performance area* can be used to determine the optimum vector of weights W_{opt} (*Wiener* filter coefficients). A quadratic performance function allows only a unique optimum global value; a local minimum does not exist. If the graphical representation is varying with many coefficients, the shape of the function will be hyper-parabolic.

The *gradient method* is used in many adaptive processes to determine the optimum vector of weights corresponding to the minimum of the performance area [6].

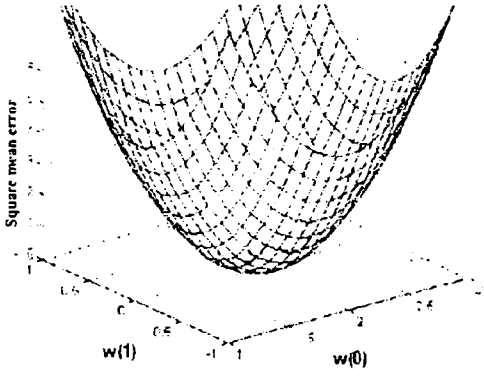

Fig. 2. Bidimensional quadratic performance area.

The gradient of the *SME* of the performance area ∇ can be obtained by derivation of (6) with respect to each component of the vector of weights:

$$\nabla = \frac{\partial \xi}{\partial W} = \left[\frac{\partial \xi}{\partial w(0)} \quad \frac{\partial \xi}{\partial w(1)} \quad \cdots \quad \frac{\partial \xi}{\partial w(N-1)}\right]^T \quad (7)$$

Note that *SME* was obtained with the probability operator of the square error (equation (6)). In the same manner, the gradient can be found by the derivation of the probabilistic operator of the square error function with respect to the vector of weights.

$$\nabla = E\left\{\frac{\partial e_k^2}{\partial W}\right\} = E\left\{2e_k \frac{\partial e_k}{\partial W}\right\} =$$
$$= -2E\left\{X_k y_k\right\} + 2E\left\{X_k X_k^T\right\} W = \quad (8)$$
$$= -2R_{yX} + 2R_{XX} W.$$

When the vector of weights (filter coefficients) has the optimum value W_{opt}, the *SME* will be minimum. So, the gradient ∇ will be zero ($\nabla = 0$). Equating (8) with zero results

$$W_{opt} = R_{XX}^{-1} R_{yX}. \quad (9)$$

Equation (9) is known as *Wiener-Hopf* equation in matrix form and the filter with coefficients given by W_{opt} represents the *Wiener* filter. For non-stationery signals W_{opt} must be computed recurrently which needs a complex computation. The *steepest descent* algorithm represents an iterative solution of the Wiener-Hopf equation.

2.2 *Steepest descent algorithm*

In practice, it is not usually to compute the optimum value W_{opt} using equation (9) because the evaluation of R_{XX}^{-1} $[N \times N]$ implies the inversion of a matrix that needs complex computations. More, if the signals are non-stationery (frequent case), the computations must be performed periodically to pursue the changes.

An alternative method to compute the optimum vector of weights W_{opt} is represented by the *steepest descent algorithm*. In according to this method, the weights are fitted recurrently with respect to gradient:

$$W_{p+1} = W_p - \mu \nabla_p, \quad (10)$$

where W_p is the weights vector at iteration p, ∇_p is the gradient vector at iteration p computed by substituti n of W_p in (8). Parameter μ is a constant that fits the size of the step and controls the stability and the convergence rate.

2.3 *The LMS algorithm*

This algorithm is very used due to its simplicity and for the easy computation. The algorithm is based on the "*steepest descent*" method, but it simplifies this method considering only one iteration per sample and computing only one *estimation* of the gradient vector ($\hat{\nabla}_k$) in each moment k.

The estimation of the gradient vector at moment k, ($\hat{\nabla}_k$), can be obtained from the error definition, (equation (3)):

$$e_k = y_k - X_k^T W \quad (11)$$

$$\hat{\nabla}_k = \left[\frac{\partial e_k^2}{\partial w(0)} \quad \frac{\partial e_k^2}{\partial w(1)} \quad \frac{\partial e_k^2}{\partial w(N-1)} \right] = 2 e_k X_k . \quad (12)$$

This gradient estimation can be replaced in equation (10) and we obtain:

$$W_{k+1} = W_k + 2\mu e_k X_k \quad (13)$$

or:

$$w_{k+1}(i) = w_k(i) + 2\mu e_k x_{k-i} , \quad (14)$$

for $i = 0,\ldots,N-1$.

Expression (14) represents the *LMS* algorithm. Parameter μ is a constant that controls the stability and the convergence rate just like in the case of *steepest descent* algorithm. According to equation (13), the hardware implementation of the *LMS* algorithm it can be made simpler, because it does not need square-, averaging- or derivation operations [6].

2.3.1 *Convergence of the LMS algorithm*

From the definition equation of the *LMS* algorithm (equation (14)), we can observe that the convergence properties of the algorithm depend on: *step size* μ; *stochastic properties* of the input signal x; N - *window length* (used number cells).

The *step size* μ: there is a time depending on optimum step $\mu_{opt}=\mu(i)$ with decreasing values for μ with time increasing (for example $\mu(0)=0.01\ldots\ \mu(\infty)=0.0001$). In practice it is very important to choose μ because that controls the convergence rate. If the value of μ is too small, it will need a longer time to converge to ξ_{min}. If the value of μ is too large, the algorithm becomes unstable and it will not converge to ξ_{min}. Generally, the *LMS* algorithm will converge if the following condition is true:

$$0 < \mu < \frac{1}{\lambda_{max}} , \quad (15)$$

where λ_{max} is the maximum value from the input covariance matrix.

The *input signal* x: can be chosen to suit the properties of the reference signal y_k.

The *length* N: a large value improves the quality of the estimations (of the convergence), but also increases the computation effort.

2.4 *The Kalman algorithm*

The *Kalman* filters, named after the scientist Rudolph E. Kalman, represents a set of mathematical equations, implementing a type of predictor-corrector estimator, that is *optimum* in the meaning of minimizing the *estimated error covariance*, when certain presumed conditions are accomplished.

2.4.1 *Discrete Kalman filter*

2.4.1.1 *Process estimation*

The *Kalman* filter tries to estimate a state $x \in \Re^n$ belonging to a controlled process, time discrete, described by the finite difference linear equation:

$$x_k = A x_{k-1} + B u_k + w_{k-1} . \quad (16)$$

with measurement $z \in R^m$:

$$z_k = H x_k + v_k . \quad (17)$$

The random variables w_k and v_k represents the process noise, respectively the measurements noise. It presumes that they are white noises, independent one to another and with normal distributions of probability:

$$\begin{aligned} p(w) &\sim N(0,Q) \\ p(v) &\sim N(0,R) \end{aligned} \quad (18)$$

In practice, *covariance of the process noise* Q and the *covariance of the measurement noise* R can be modified in each moment, but in our case it is presumed that they are constants.

Matrix $A[n\times n]$ from (16) describes the process state from the previous moment $k-1$ at the current moment k, in default of driver function or the process noise. In practice, matrix A can be modified at each moment, but here it is presumed to be constant. Matrix $B[n\times 1]$ describes the optional control of input $u \in \Re^l$ about the state x. Matrix $H[m\times n]$ from equation, intended to measure (17), describes the measurement z_k. In practice, matrix H can be modified at each moment or at each measurement, but here it is presumed to be constant [4].

2.4.1.2 *Filter computation*

$\hat{x}_k^- \in \Re^n$ is defined to be the *a priori* estimated state at moment k and $\hat{x}_k \in \Re^n$ the *a posteriori* estimated state at moment k having the measurement z_k. It can be defined now the *a priori* and *a posteriori* estimated errors:

$$e_k^- \equiv x_k - \hat{x}_k^- \quad \text{and} \quad e_k \equiv x_k - \hat{x}_k . \quad (19)$$

Under these conditions, the covariances of the *a priori* and the *a posteriori* estimated error, is:

$$P_k^- = E\left[e_k^- e_k^{-T} \right] \quad (20)$$

respectively

$$P_k = E\left[e_k e_k^T \right] . \quad (21)$$

To obtain the equations for the *Kalman* filter, we have to find an equation which computes the *a posteriori* estimated state $\hat{x}_k$ as a linear combination of the *a priori* estimated state $\hat{x}_k^-$ and of the weighted difference between the current measurement z_k and the measurement prediction $H\hat{x}_k^-$:

$$\hat{x}_k = \hat{x}_k^- + K(z_k - H\hat{x}_k^-). \quad (22)$$

The term $(z_k - H\hat{x}_k^-)$ from equation (22) is called the measurement *innovation* or *residue*. This difference reflects the discrepancy between the measurement prediction $H\hat{x}_k^-$ and the current measurement z_k. If the residue is zero then the two quantities are equal.

The matrix $K[n \times m]$ from (22) represents the gain or the interference factor and its role is to minimize the *a posteriori* estimated error covariance P_k from (21). This can be achieved by substituting equation (22) in the definition equation of e_k; the obtained error will be replaced in (21) and the probability operator will be computed. The obtained result will be derived with respect to K, equated with zero and solved to compute K. The most encountered form of the *Kalman* gain, which minimizes equation (21), is:

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} = \frac{P_k^- H^T}{HP_k^- H^T + R} \quad (23)$$

When the covariance of the measurement noise R tends to zero, the gain K weights better the residue:

$$\lim_{R_k \to 0} K_k = H^{-1}. \quad (24)$$

When the *a priori* estimated error covariance P_k^- is achieving zero. the gain K weights very slightly the residue:

$$\lim_{P_k^- \to 0} K_k = H^{-1}. \quad (25)$$

So, when the measurement noise covariance R is achieving zero, the current measurement z_k matters more and the measurement prediction $H\hat{x}_k^-$ matters less. On the other side, when the *a priori* estimated error covariance P_k^- is achieving zero, the current measurement z_k matters less and the measurement prediction $H\hat{x}_k^-$ matters more.

2.4.1.3 *Description of the Kalman algorithm*

The *Kalman* filter estimates a process using a feedback control: the filter estimates the state of a process at a moment and then obtains a feedback in the form of the measurements (in noisy conditions). Thus. equations of the *Kalman* filter are divided in two groups: *equations for re-update in time* and *equations for re-update of measurement*. The equations for re-update in time are responsible for time designing of the current state and the estimated error covariance to obtain the *a priori* estimations for the next instant. The equations for re-update of the measurements are responsible for the realization of the inverse feedback – the inclusion of new measurement in the *a priori* estimation to obtain an improved *a posteriori* estimation. The equations for re-update in time are also called *prediction equations* and the equations for re-update of measurement are also called the *correction equations*. So, the final estimating algorithm is a *predictor-corrector* algorithm, shown in figure 3.

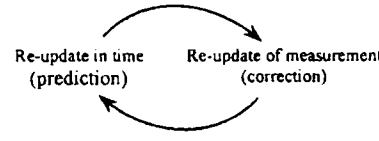

Fig. 3. Discrete *Kalman* filter cycle.

The specific time and measurements re-update equations are presented below:

$$\begin{cases} \hat{x}_k^- = A\hat{x}_{k-1}^- + Bu_k \\ P_k^- = AP_{k-1}A^T + Q \end{cases} \quad (26)$$

$$\begin{cases} K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \\ \hat{x}_k = \hat{x}_k^- + K(z_k - H\hat{x}_k^-) \\ P_k = (1 - K_k H)P_k^- \end{cases} \quad (27)$$

From (27) we notice that the first step consists in *Kalman* gain K_k computation. The next step consists in measurement updating to obtain z_k and generating the *a posteriori* estimated state. The final step consists in obtaining the *a posteriori* estimated error covariance.

After each re-update of the time and measurements pairs, the process is repeated using the *a posteriori* estimations to compute the new *a priori* estimations. That recurrence is one of the most interesting property of the *Kalman* filters, that makes its practical implementation to be more easier to realize relative to implementation of the *Wiener* filter, designed to work directly on all data for each estimation [4]. Instead, the *Kalman* filter computes recursively the current estimation based on all measurements performed.

3. IMPLEMENTATION AND SIMULATION

In the last years, LabVIEW and Matlab's Simulink became the most well known software packages used in education and industry for modeling and simulation of dynamic systems.

An example for processing signals using the *LMS* algorithm is shown in figure 4. The *LMS* algorithm structure with 4 coefficients is shown in figure 5.

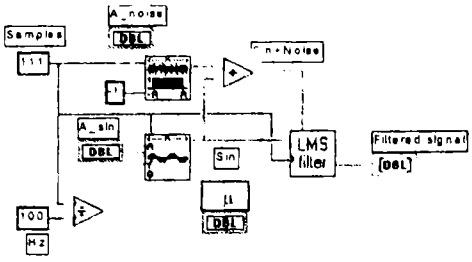

Fig. 4. Signal processing with *LMS* algorithm

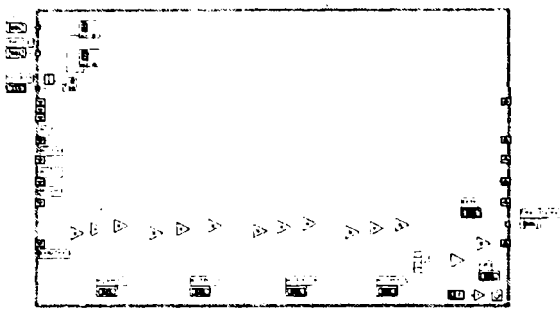

Fig. 5. The *LMS* algorithm structure

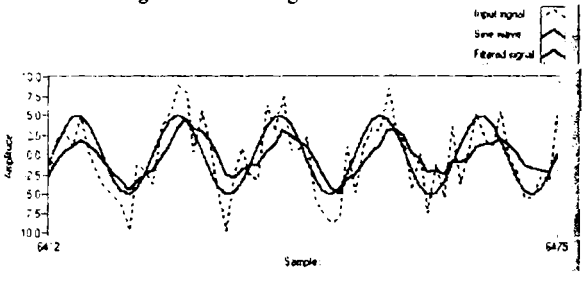

Fig. 6. Input signal, Filtered signal; Desired signal (Sine wave)

The results in figure 6 were obtained with the following program specifications: the algorithm contains 4 coefficients; amplitude of sine wave = *10V*; frequency of sine wave = *100Hz*; number of samples for sine wave = *111*; amplitude of noise (white noise) wave = *10V*; step value $\mu=5\cdot10^{-7}$.

In figure 7 are shown the results obtained using the same specifications, but the number of the filter coefficients was increased to 16. It is visible that the filtered signal (continuous black line) is closer to the desired signal (continuous gray line) than in figure 6.

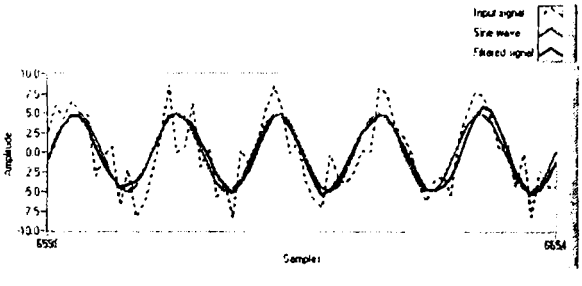

Fig. 7. Input signal, Filtered signal; Desired signal (Sine wave).

The results of the *LMS* algorithm – figure 9 (implemented in Simulink – figure 8) were obtained with the following program specifications: *LMS* filter with 16 coefficients; the step $\mu=5\cdot10^{-2}$; the input signal is an audio wave signal on 16 bits, with *8 KHz* frequency, on single channel (mono).

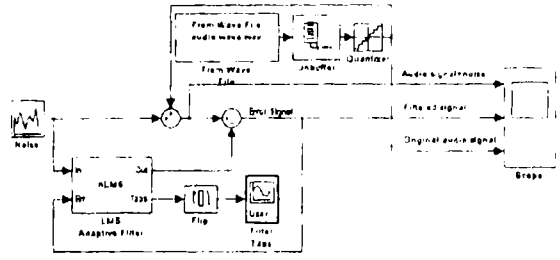

Fig. 8. Signal processing with adaptive *LMS* algorithm

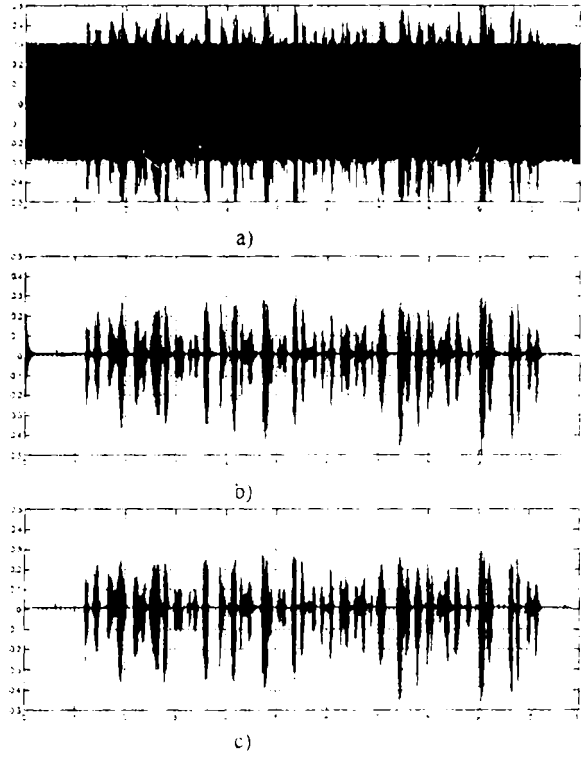

Fig. 9. a) Audio signal with noise, b) filtered audio signal; c) desired audio signal

The results of the filtering example (figure 11) with the *Kalman* algorithm (implemented in LabVIEW – figure 10) were obtained under the following program specifications: amplitude of sine wave = *10V*; noise amplitude = *10V*; sine wave frequency = *100Hz*; samples of sine = *111*; $Q=5\cdot10^{-3}$; $R=1\cdot10^{-2}$. Figure 10 shows the structure of the *Kalman* filter.

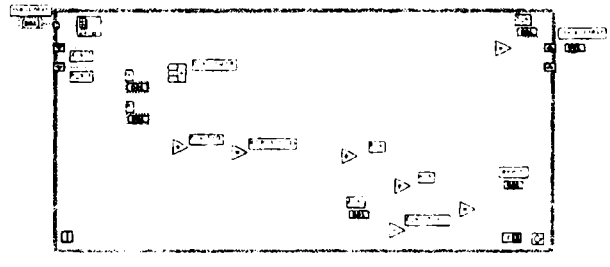

Fig. 10. *Kalman* filter structure

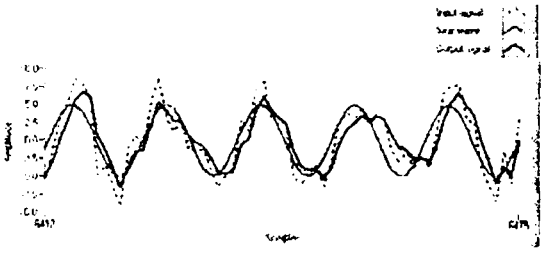

Fig. 11. Input signal, Filtered signal, Desired signal (Sine wave)

In the simulation example with *Kalman* algorithm (implemented in Simulink), an input audio signal of *8 KHZ* frequency, represented on 16 bits, and $Q=1\cdot10^{-8}$; $R=1\cdot10^{-2}$ were considered. The obtained results are shown in figure 12.

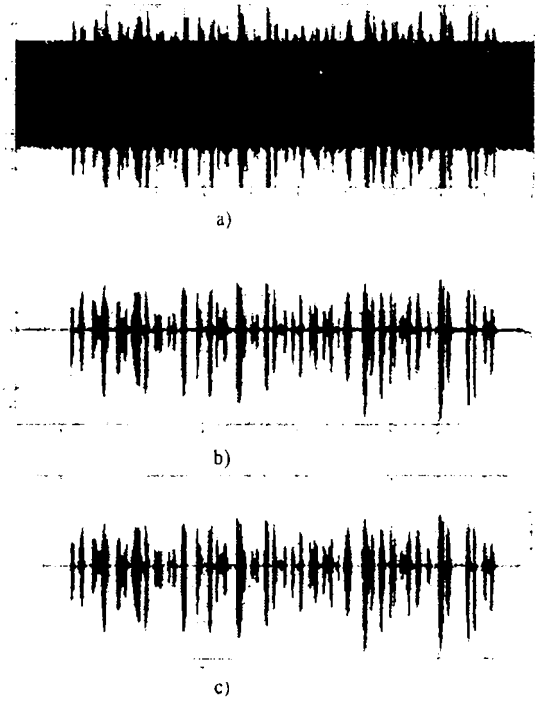

Fig. 12. a) Audio signal with noise, b) filtered audio signal; c) desired audio signal.

4. CONCLUSIONS

A. *General conclusions.*

In this paper, several techniques for designing and implementing adaptive filters were presented. These techniques were based on the gradient algorithm, being the simplest and the most efficient instrument for varying the coefficients.

The gradient algorithm leads to slowly modifying filter coefficients values, when it requests a reduced residual error and it is used in the simpler form (the *sign algorithm*). To find the most rapid adaptation rate, all of the coefficients can be re-computed periodically using rapid iterative procedures.

It is possible to consider some other criteria, which, for other precision applications, are more suitable than the minimizing *SME* criterion, thus, more efficient algorithms than the *gradient algorithm* can be developed. The *gradient algorithm* can be improved, for example, using different coefficient step variations, which are obtained from statistical estimations of the signal characteristics. However, due to implementation imperfections, applying these algorithms and sensitivity problems become difficult.

The specific noise cancellation case was already studied since 4 decades but lately hardware implementation possibilities of the theoretical systems with signal processors are loomed.

This paper tries to achieve some important sides of the adaptive systems in noise cancellation.

B. *Conclusions concerning the simulation results.*

From simulations, we can observe when the noise amplitude is growing up, the filtered signal is visible distorted, that reduces the respectively algorithm performances. Otherwise, increasing the number of filter coefficients, the accuracy of the filtered signal increases.

Another important problem is choosing the optimum parameters of the adaptive filters. For example, in case of the *LMS* algorithm, choosing the step (μ), which determines the convergence rate is critical. If μ is too large, the algorithm will converge very rapidly but will present oscillations until stability limit is reached, or, the effect of inverse error minimizing - another drawback - appears. If a too small step μ is chosen, oscillations will not appear during the convergence process, but the convergence speed is slower.

The optimum *Wiener* filter theory was made for random stationery processes. When the statistical properties of the random processes are changing in time, the above description becomes more difficult. Due to the permanently modifying of the error surface of which minimum is to be searched, the adaptive algorithm must ensure not only the convergence to the optimum solution, but also to follow the continuous changing of this optimum value. The *Kalman* filter theory that allows a model, for the considered application, based on state equations gives the solution. The obtained recursive algorithm is more rapid than the *LMS* algorithm and less dependent by the static characteristics of input data, but presumes more complex computations.

5. REFERENCES

[1] Adelaida Mateescu, Neculai Dumitriu, Lucian Stanciu, "Semnale şi Sisteme Aplicaţii în filtrarea semnalelor", Ed. Teora, 2001;

[2] Maurice Bellanger, "Digital Processing of Signals. Theory and Practice", Ed. John Wiley&Sons, 1984;

[3] J.G. Proakis, "Advanced Digital Signal Processing", Ed. McGraw – Hill, 1992;

[4] Greg Welch, Garry Bishop, "An Introduction to the Kalman Filter", Ed.ACM, 2001; (http://www.cs.unc.edu/~welch/kalman);

[5] Yifen Tu, "Multiple Reference Active Noise Control", March 1997;

[6] Digital Signal Processing - Chapter 7: Adaptive Filtering (http://www.staff.ncl.ac.uk/oliver.hinton/eee305/Chapter7.pdf);

[7] Mathlab Help *Signal Processing Toolbox, Simulink.*