Performance Comparison of Adaptive Algorithms for Adaptive line Enhancer

Sanjeev Kumar Dhull* Sandeep K. Arya* and O.P.Sahu#
*Guru Jambheshwar University of Science and Technology, Hisar, #National Institute of Technology, Kurukshetra

Abstract
We have designed and simulated an adaptive line enhancer system for conferencing. This system is based upon a least-mean-square (LMS) and recursive adaptive algorithm (RLS) Performance of ALE is compared for LMS&RLS algorithms.
Keywords:LMS,RLS,ALE

1. INTRODUCTION

Among the noise cancelling applications of the adaptive filters are the adaptive noise canceller and adaptive line enhancer. While the adaptive noise canceller uses a contaminated input, as well as a reference input correlated with the buried signal, the adaptive Line enhancer uses a single input signal. A delayed version of the single input signal is used to decorrelate the noise component and remove it from the device output. There are a number of circumstances where a periodic signal is corrupted by broad-band noise, i.e. the Noise with components over a wide range of frequencies. This case utilizes an Adaptive Line Enhancer, for detection and tracking of the periodic components buried in broad band noise. In ALE, a fixed delay is inserted in a reference input drawn directly from the input signal, so as to cancel the interference. The delay chosen must be of sufficient length to cause the broad-band signal components in the reference input to become de-correlated from those in the primary input.

1.1.Different Issues of Adaptive Line Enhancer

An ALE consists of a delay element .The input x_n consists of signal s_n plus noise n_n. The estimated output $\widehat{x_n}$ is subtracted from the input signal x_n to produce the estimation error e_n. This estimation error is in turn used to adaptively control the filter. The filter input $x_{n-\Delta}$ which is the input signal delayed with Δ samples [16]. The delay has to be chosen such that the noise in the original input signal x_n and in the delayed filter input $x_{n-\Delta}$ is uncorrelated, so that it can be suppressed by the filter. This filter is an adaptive filter whose tap weights are controlled by an adaptive algorithm. The filter operates by iteratively minimizing the quadratic index using an instantaneous estimate of the gradient of the quadratic performance surface to update the filter coefficients. Thus ALE refers to the case where a noisy signal, x_n, consisting of a sinusoidal component, s_n, is available and the requirement is to remove the noise part of the signal, n_n.It consists of a de-correlation stage, symbolized by delay, Δ. The de-correlation stage attempts to remove any correlation that may exist between the samples of noise, by shifting the samples Δ apart. As a result, the predictor can only make a prediction about the sinusoidal component of x_n, and when adapted to minimize the instantaneous squared error output, e, the line enhancer will be a filter optimized (the Wiener solution) or tuned to the sinusoidal component Operation of the adaptive line enhancer can be understood intuitively as follows. The delay causes de-correlation between the noise components of the input data in the two channels while introducing a simple phase difference between the sinusoidal components. The adaptive filter responds by forming a transfer function equivalent to that of a narrow-band filter centered at the frequency of the sinusoidal components. The noise component of the delayed input is rejected, while the phase difference of the sinusoidal components is readjusted so that they cancel each other at the summing junction, producing a minimum error signal composed of the noise component of the instantaneous input data alone. Signal x_n, is available which is contaminated by noise. In such a case, the signal x_n, provides its own reference signal $x_{n-\Delta}$, which is taken to be a delayed replica of x_n, that is $y_n = x_{n-\Delta}$.The adaptive filter will respond by canceling any components of the main signalx_n that are in any way correlated with the secondary signal $y_n = x_{-\Delta}$. Suppose the signal x_n consists of two parts: a narrowband component that has long-range correlations such as a sinusoid, and a broadband component which will tend to have short-range correlations [13]. One of these could represent the desired signal and the other an undesired interfering noise. Pictorially the autocorrelations of the two components could look as follows. Where k_{NB} and k_{BB} are effectively the self-correlation lengths of the narrowband and broadband components, respectively; beyond these lags, the respective correlations die out quickly. Suppose the delay Δ is selected so that

$$k_{BB} \leq \Delta \leq k_{NB}$$

Since Δ is longer than the effective correlation length of the BB component, the delayed replica BB (t − Δ) will be entirely uncorrelated with the BB part of the main signal. The adaptive filter will not be able to respond to this component. On the other hand, since Δ is shorter than the correlation length of the NB component, the delayed replica NB (t−Δ) that appears in the secondary input will still be correlated with the NB part of the main signal, and the filter will respond to cancel it [13]. This paper is organized in four sections. Section II describes algorithms which are to be compared. Further, section III Discuss the results. In the end section IV conclude the paper.

2. ADAPTIVE ALGORITHMS

Adaptive algorithms can be divided into various categories. Two among them is explained as following

2.1. Least Mean Square (LMS) Algorithm

The Least Mean Square (LMS) algorithm was first developed by Widrow and Hoff in 1959. Since then it has become one of the most widely used algorithms in adaptive filtering. The LMS algorithm is a type of adaptive filter known as stochastic gradient-based algorithms as it utilizes the gradient vector of the filter tap weights to converge on the optimal wiener solution. It is well known and widely used due to its computational simplicity. It is this simplicity that has made it the benchmark, against which all other adaptive filtering algorithms are judged [7], with each iteration, the filter tap weights of the adaptive filter are updated according to the following formula

$$w(n+1) = w(n) + 2\mu e(n)x(n) \quad (1)$$

Here **x**(n) is the input vector of time delayed input values, **x**(n) = [x(n) x(n-1) x(n-2) .. x (n-N+1)]^T. The vector **w**(n) = [w0(n) w1(n) w2(n) .. wN-1(n)] ^{T} represents the coefficients of the adaptive FIR filter tap weight vector at time n [11]. The parameter μ is known as the step size parameter and is a small positive constant. This step size parameter controls the influence of the updating factor. Selection of a suitable value for μ is imperative to the performance of the LMS algorithm, if the value is too small the time the adaptive filter takes to converge on the optimal solution will be too long; if μ is too large the adaptive filter becomes unstable and its output diverges. The derivation of the LMS algorithm builds upon the theory of the wiener solution for the optimal filter tap weights, w_o. It also depends on the steepest-descent algorithm [11]. This is a formula which updates the filter coefficients using the current tap weight vector and the current gradient of the cost function with respect to the filter tap weight coefficient vector, ∇ξ(n).

$$w(n+1) = w(n) - \mu\nabla\xi(n) \quad (2)$$

Where $\xi(n)$=E [e^2 (n)]

As the negative gradient vector points in the direction of steepest descent for the N dimensional quadratic cost function, each recursion shifts the value of the filter coefficients closer toward their optimum value, which corresponds to the minimum achievable value of the cost function, ξ(n).The LMS algorithm is a random process implementation of the steepest descent algorithm, from equation 2. Here the expectation for the error signal is not known so the instantaneous value is used as an estimate. The steepest descent algorithm then becomes equation 3.

$$w(n+1) = w(n) - \mu\nabla\xi(n) \quad (3)$$

Where $\xi(n) = e^2(n)$

The gradient of the cost function, ∇ξ(n), can alternatively be expressed in the following form:

$$\begin{aligned}\nabla\xi(n) &= \nabla(e^2(n)) \\ &= e(n)\frac{\partial e(n)}{\partial w} \\ &= 2e(n)\frac{\partial(d(n)-y(n))}{\partial w} \\ &= -2e(n)\frac{\partial e w^T(n)x(n)}{\partial w} \\ &= -2e(n)x(n) \quad (4)\end{aligned}$$

Substituting this into the steepest descent algorithm of equation 2, we arrive at the recursion for the LMS algorithm

$$w(n+1) = w(n) + 2\mu e(n)x(n) \quad (5)$$

2.2. Implementation of the LMS algorithm:

Each iteration of the LMS algorithm requires the following distinct steps in the given order:

1. The output of the filter, y(n) is calculated using the equation (6):

$$y(n) = \sum_{i=0}^{N-1} w(n)xx(n-i) = w^T(n)x(n) \quad (6)$$

2. The value of the error estimation is calculated using equation (7)

$$e(n) = d(n) - y(n) \quad (7)$$

3. The tap weights of the FIR vector are updated in preparation for the next iteration by the equation (8)

$$w(n+1) = w(n) + 2\mu e(n)x(n) \quad (8)$$

The main reason for the LMS algorithms popularity in adaptive filtering is its computational simplicity, making it easier to implement than all other commonly used adaptive algorithms. For each iteration the LMS algorithm requires 2N additions and 2N+1 multiplications (N for calculating the output, y(n), one for 2μe(n) and an additional N for the scalar by vector multiplication).

2.3. Recursive Least Squares (RLS) Algorithm

The other class of adaptive filtering techniques is known as Recursive Least Squares (RLS) algorithms. They may be used in place of the LMS algorithm in any adaptive filtering application. [13].The filter weights are optimal at each time instant n. Their main disadvantage is that they require a fair amount of computation. These algorithms attempt to minimize the cost function .. With values of λ<1 more importance [11] is given to the most recent error estimates and thus the more recent input samples, this results in a scheme that places more emphasis on recent samples of observed data and tends to forget the past .

$$\zeta(n) = \sum_{k=1}^{n} \lambda^{n-k} e_n^2(k) \quad (9)$$

Unlike the LMS algorithm and its derivatives, the RLS algorithm directly considers the values of previous error estimations. RLS algorithms are known for excellent performance when working in time varying environments. These advantages come with the cost of an increased computational complexity and some stability problems. The RLS cost function of equation (9) shows that at a time n, all previous values of the estimation error since the commencement of the RLS algorithm are required. Clearly as time progresses the amount of data required to process this algorithm

increases. The fact that memory and computation capabilities are limited makes the RLS algorithm a practical impossibility in its purest form. However, the derivation still assumes that all data values are processed. In practice only a finite number of previous values are considered, this number corresponds to the order of the RLS FIR filter, First we define $y_n(k)$ as the output of the FIR filter, at n, using the current tap weight vector, and the input vector of a previous time k. The estimation error value $e_n(k)$ is the difference between the desired output value at time k, and the corresponding value of $y_n(k)$. These and other appropriate definitions are expressed in equation 2, for k=1, 2, 3,., n.

$$\begin{aligned} y_n(k) &= w^T(n)x(k) \\ e_n(k) &= d(k) - y_n(k) \\ d(n) &= [d(1), d(2) \ldots . d(n)]^{\mathrm{T}} \\ y(n) &= [y_n(1), y_n(2) \ldots y_n(n)]^{\mathrm{T}} \\ e(n) &= [e_n(1), e_n(2) \ldots . e_n(n)]^{\mathrm{T}} \\ e(n) &= d(n) - \ y(n) \end{aligned} \quad (10)$$

If we define the X(n) as the matrix consisting of the n previous input column vector up to the present time then y(n) can also be expressed as equation (3)

$$\begin{aligned} X(n) &= [x(1), x(2), \ldots \ldots x(n)] \\ y(n) &= X^T(n)w(n) \end{aligned} \quad (11)$$

The cost function of equation (1) can then be expressed in matrix vector form using Λ (n), a diagonal matrix consisting of the weighting factors.

$$\begin{aligned} \zeta(n) &= \sum_{k=1}^{n} \lambda^{n-k} e_n^2(k) \\ &= e^T(n)\tilde{\Lambda}(n)e(n) \end{aligned}$$

$$\text{Where } \tilde{\Lambda}(n) = \begin{bmatrix} \lambda^{n-1} & 0 & 0 & \ldots & 0 \\ 0 & \lambda^{n-2} & 0 & \ldots & 0 \\ 0 & 0 & \lambda^{n-3} & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & 1 \end{bmatrix} \quad (4)$$

Substituting values from equations (2) and (3), the cost function can be expanded then reduced as in equation (5) (Temporarily dropping (n) notation for clarity).

$$\begin{aligned} \zeta(n) &= e^T(n)\tilde{\Lambda}(n)e(n) \\ &= d^T\tilde{\Lambda}d - d^T\tilde{\Lambda}y - y^T\tilde{\Lambda}d + y^T\tilde{\Lambda}y \\ &= d^T\tilde{\Lambda}d - d^T\tilde{\Lambda}(X^Tw) - (X^Tw)^T\tilde{\Lambda}d \\ &\quad + (X^Tw)^T\tilde{\Lambda}(X^Tw) \\ &= d^T\tilde{\Lambda}d - 2\tilde{\theta}_\lambda^T w + w^T\tilde{\psi}_\lambda w \end{aligned}$$

$$\text{Where } \tilde{\psi}_\lambda(n) = X(n)\tilde{\Lambda}(n)X^T(n) \text{ , and } 2\tilde{\theta}_\lambda^T = X(n)\tilde{\Lambda}(n)d(n) \quad (12)$$

We then derive the gradient of the above expression for the cost function with respect to the filter tap weights. By forcing this to zero we then find the coefficients for the filter, w(n) which minimizes the cost function.

$$\tilde{\psi}_\lambda(n)\bar{w}(n) = 2\tilde{\theta}_\lambda(n)$$

The matrix Ψ(n) in the above equation can be expanded and then rearranged in a recursive form, [11] The matrix Ψ(n) in the above equation can be expanded and then rearranged in a recursive form, we can then use the special form of the matrix inversion lemma to find an inverse for this matrix, which is required to calculate the tap weight vector update. The vector **k**(n) is known as the gain vector and is included in order to simplify the calculation.

$$\begin{aligned} \tilde{\psi}_\lambda^{-1}(n) &= \lambda\tilde{\psi}_\lambda^{-1}(n-1) + x(n)x^T(n) \\ &= \lambda^{-1}\tilde{\psi}_\lambda^{-1}(n-1) \\ &\quad - \frac{\lambda^{-2}\tilde{\psi}_\lambda^{-1}(n-1)x(n)x^T(n)\tilde{\psi}_\lambda^{-1}(n-1)}{1+\lambda^{-1}x^T(n)\tilde{\psi}_\lambda^{-1}(n-1)x(n)} \\ &= \lambda^{-1}(\tilde{\psi}_\lambda^{-1}(n-1) - k(n)x^T(n)\tilde{\psi}_\lambda^{-1}(n-1)) \\ where\ k(n) &= \frac{\lambda^{-1}\tilde{\psi}_\lambda^{-1}(n-1)x(n)}{1+\lambda^{-1}x^T(n)\tilde{\psi}_\lambda^{-1}(n-1)x(n)} \\ &= \tilde{\psi}_\lambda^{-1}(n)x(n) \end{aligned} \quad (13)$$

The vector $\theta_\lambda(n)$ can also be expressed in a recursive form. Using this we can finally arrive at the filter weight update vector for the RLS algorithm, as in equation (14).

$$\begin{aligned} \widetilde{\theta_\lambda}(n) &= \lambda\widetilde{\theta_\lambda}(n-1) + x(n)d(n) \\ \bar{w}(n) &= \tilde{\psi}_\lambda^{-1}(n)\widetilde{\theta_\lambda}(n) \\ &= \tilde{\psi}_\lambda^{-1}(n-1)\widetilde{\theta_\lambda}(n-1) \\ &\quad - k(n)x^T\tilde{\psi}_\lambda^{-1}(n-1)\widetilde{\theta_\lambda}(n-1) \\ &\quad + k(n)d(n) \\ &= \bar{w}(n-1) - k(n)x^T(n)\bar{w}(n-1) + k(n)d(n) \\ &= \bar{w}(n-1) + k(n)(d(n) - \bar{w}^T(n-1)x(n)) \\ \bar{w}(n) &= \bar{w}(n-1) + k(n)\overline{e_{n-1}}(n) \end{aligned}$$

$$where\ \overline{e_{n-1}}(n) = d(n) - \bar{w}^T(n-1)x(n) \quad (14)$$

2.4. Implementation of the RLS algorithm:

The memory of the RLS algorithm is confined to a finite number of values, corresponding to the order of the filter tap weight vector. Firstly, two factors of the RLS implementation should be noted: the first is that although matrix inversion is essential to the derivation of the RLS algorithm, no matrix inversion calculations are required for the implementation, thus greatly reducing the amount of computational complexity of the algorithm. Secondly, unlike the LMS based algorithms, current variables are updated within the iteration they are to be used, using values from the previous iteration. To implement the RLS algorithm, the following steps are executed in the following order.

1. The filter output is calculated using the filter tap weights from the previous iteration and the current input vector.
$$\bar{y}_{n-1}(n) = \bar{w}^T(n-1)x(n)$$
2. The intermediate gain vector is calculated using equation
$$u(n) = \tilde{\psi}_\lambda^{-1}(n-1)x(n)$$
$$k(n) = \frac{1}{\lambda + x^T(n)u(n)}u(n)$$
3. The estimation error value is calculated using equation
$$\overline{e_{n-1}}(n) = d(n) - \bar{y}_{n-1}(n)$$

4. The gain vector calculated in equation

$$\overline{w}(n) = \overline{w}(n-1) + k(n)\overline{e_{n-1}}(n)$$

Each iteration of the RLS algorithm requires $4N^2$ multiplication operations and $3N^2$ additions [7].This makes its very costly to implement, thus LMS based algorithms, while they do not perform as well, are more favorable in practical situations.

3. SIMULATIONS RESULTS OF THE ALE

The comparative analysis of different algorithms is done by comparing the simulation results of different models which are fed with same input signals but are based on adaptive filters with different algorithms. Here, we describe the simulation modeling used for the generating the analysis. Each of the adaptive filtering algorithms outlined in last section were implemented using the simulink models designed.

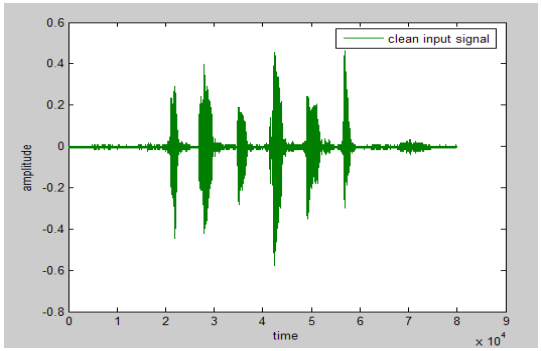

Figure 1.1 Clean input speech signal

We have designed different models using these two adaptive filters, and simulated. The input speech signal is shown in the figure 1.1.For the simulation we have taken the Gaussian noise signal, with zero mean and unit variance and the sampling time 1/8000. The noise is plotted in the in the figure 1.2.

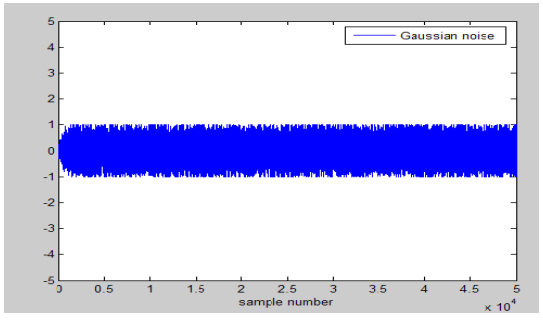

Figure 1.2 The Gaussian noise signal with zero mean and unit variance.

The figures from 1.4 to 1.7 shows the filtered signals obtained as the outputs from the LMS ALE and RLS ALE models, with filter length 40 for all the models. The signal used for this simulation is a speech signal as shown in the figure 1.3, along with the signal when contaminated by Gaussian noise. The convergence parameter used in the LMS algorithm is 0.002.

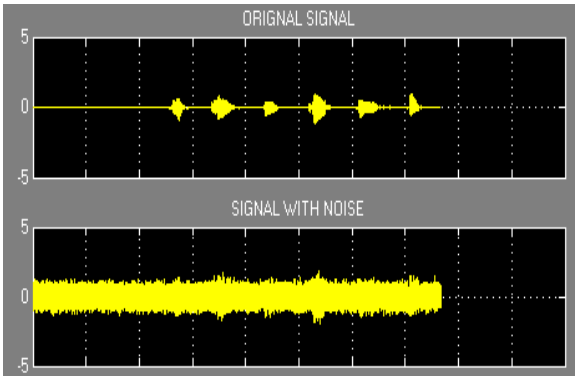

Figure 1.3 Clean input signal and signal corrupted with noise

For the simulation, we have taken consecutive five values of delay parameter, viz. (Δ=1, 2…5). The obtained outputs with five different delays for both the LMS and RLS ALE for the given signal are plotted in the figure 1.4 to 1.8.

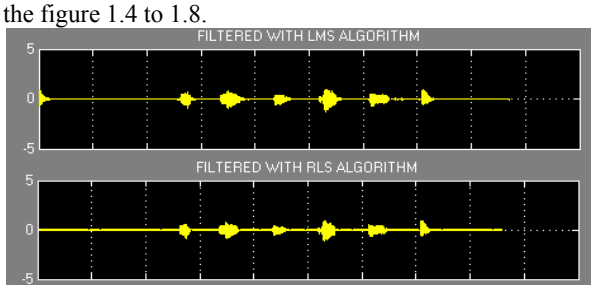

Figure 1.4 Comparison of the filtered signal with the LMS and RLS ALE for Δ= 1

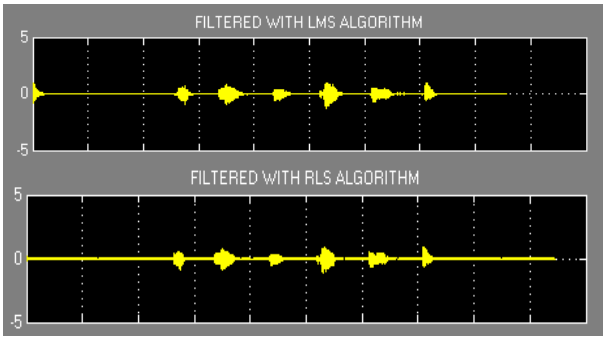

Figure 1.5 Comparison of the filtered signal with the LMS and RLS ALE for Δ=2

.

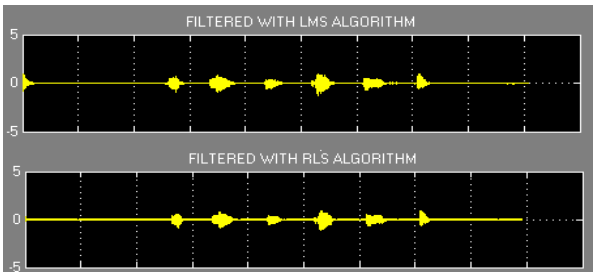

Figure 1.6 Comparison of the filtered signal with the LMS and RLS ALE for Δ=3

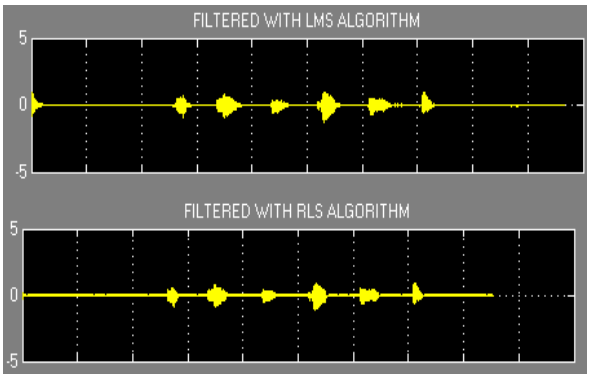

Figure 1.7 Comparison of the filtered signal with the LMS and RLS ALE for Δ=4

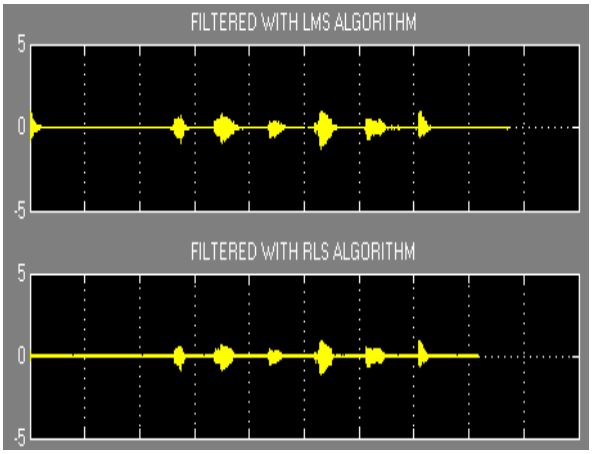

Figure 1.8 Comparison of the filtered signal with the LMS and RLS ALE for Δ=5

From the above simulations we obtained the signals filtered with LMS and RLS filters with different values of delay. In these simulated results, we observe that the output signal obtained with the LMS ALE is somewhat inferior to the signal obtained with the RLS ALE. We then plotted the graph between the SNR's of the filtered signals for particular ALE with respect to the varying values of the time delay. The SNR plots obtained for the LMS and RLS filters are shown in the figure 1.9 and 1.10 respectively. The delay is taken along the x-coordinate and the SNR along the y-coordinate.

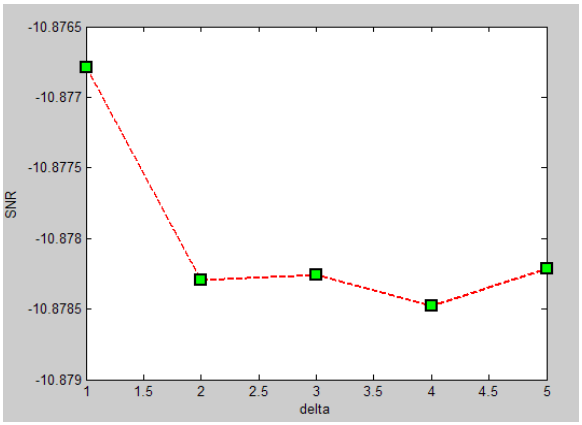

Figure 1.9 SNR plot for LMS ALE

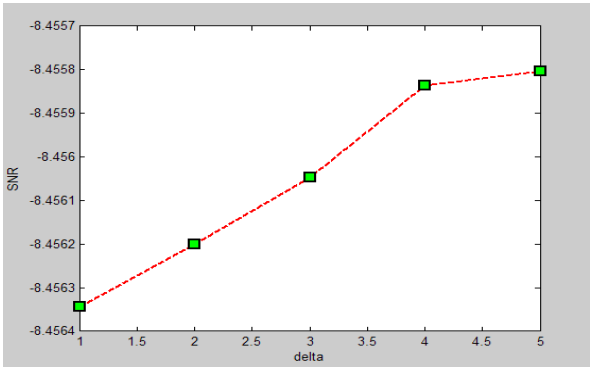

Figure 1.10 SNR plot for RLS ALE

When the delay is taken as one, for the LMS ALE, the obtained SNR is -10.870dB . The SNR in RLS ALE is approx. -8.4564 dB. Thus the SNR is inferior in case of LMS ALE when compared to RLS ALE.

4. CONCLUSION.

The performance of LMS ALE and RLS ALE are compared on the basis of the value of SNR of output signal. The simulations illustrated that the LMS ALE performance is inferior to that of the RLS ALE. The RLS ALE showed higher SNR value as in comparison to that of the LMS ALE. But in real time we prefer the LMS ALE over the RLS ALE, since because of its simplicity i.e. low computational cost, ease of implementation of the LMS algorithm and since it is robust and reliable, usually we go for LMS ALE. The RLS algorithm offer advantages over LMS algorithm such as faster convergence, however the RLS algorithm has the disadvantage of higher complexity which makes the implementation and analysis arduous.

REFERENCES

1. B. Widrow et al., "Adaptive noise canceling principles and applications," Proceedings of the IEEE, vol. 63, pp. 1692-1716, December 1975.

2. Mukund Padmanabhan, “A Hyper-stable Adaptive Line Enhancer for Fast Tracking of Sinusoidal Inputs,” ieee transactions on circuits and systems of analog and digital signal processing, vol44-april 1996.
3. H.G. Yeh, T. M. Nguyen “Adaptive Line Enhancers for Fast Acquisition,” TDA Progress Report 42-119 November 15, 1994
4. Mounir Ghogho, Mohamed Ibnkahla, and Neil J. Bershad ,“Analytic Behavior of the LMS adaptive Line Enhancer for Sinusoids Corrupted by Multiplicative and Additive Noise,” ieee transactions on signal processing,vol.46,no.9,sept.
5. Jafar Ramadhan Mohammed,“A New Simple Adaptive Noise Cancellation Scheme Based On ALE and NLMS Filter,” Fifth Annual Conference on Communication Networks and Services Research (CNSR'07) ieee-2007.
6. Naoto Sasaokal, Yoshio Itohl, Keiichi Wakizaka and Kensaku Fujii, “ a study on less computational load of noise reduction method based on noise reduction method based on Ale and noise estimation filter,” Proceedings of 2005 International Symposium on Intelligent Signal Processing and Communication Systems, December 13-16, 2005.
7. Farhang-Boroujeny, B. 1999, Adaptive Filters, Theory and Applications. John Wiley and Sons, New York
8. GUO Yemi and ZHAO Junwei ,“Combined Kurtosis Driven Variable Step Size Adaptive Line Enhancer,” 2004 8th International Conference on Control, Automation, Robotics and Vision Kunming. China, 6-9th December 2004.
9. Toshihiro Miyawaki, Naoto Sasaoka, Yoshio Itoh, Kensaku Fujiit and Sumio Tsuiki ,“A Study on Pitch Detection of Sinusoidal Noise for Noise Reduction System,” 2006 international Symposium on Intelligent Signal Processing and Communication Systems (ISPACS2006).
10. Isao Nakanishi and Yuudai Nagata, Yoshio Itoh, Yutaka Fukui ,“Single-Channel Speech Enhancement Based on Frequency Domain ALE” ISCAS 2006
11. Michael Hutson” Acoustic echo cancellation using DSP”Nov.2003
12. P.M.Clarkson“Adaptive and optimal signal processing,” CRC,1993
13. Sophocles J. Orfanidis,,“Optimum Signal Processing,"Second Edition Mc-Graw Hill,1988
14. Naoto Sasaoka, Koji Shimada, Shota Sonobe, Yoshio Itoh and Kensaku Fujii ,“speech enhancement based on adaptive filter with variable step size for wideband and periodic noise,” 2009 ,IEEE
15. V.Umapathi Reddy,Bo Egardt, Thomas Kailath “Optimized lattice-form Adaptive Line Enhance for a sinusoidal signal in Broad band noise,”1981,ieee
16. Farida Sandberg, Ulrike Richter,“Adaptive signal processing, lab adaptive line enhancer,”2007.
17. A. Nehorai and M. Morf,, “Enhancement of sinusoids in colored noise and the whitening performance of exact least squares predictors,” ieee Trans. Accost., Speech, Signal Processing, vol. ASSP-30, pp. 353–363, June 1982.
18. D. W. Tufts, L. J. Griffiths, B. Widrow, J. Glover, J. McCool, and J.Treichler,“Adaptive Line Enhancement and Spectrum Analysis," Proceedings of the IEEE, Letter, pp. 169, January 1977.
19. J. Treichler, "Transient and convergent behavior of the adaptive line enhancer," ieee Trans. Acoust., Speech, Signal Processing, Vol. ASSP-27, No. 1, pp. 53-63, February 1979.
20. J. 1. Rickard and J. R. Zeidler, "Second-order output statistics of the adaptive line enhancer," IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-27, no. 1, pp. 31-39, February 1979.
21. N. J. Bershad and O. Macchi, “Adaptive recovery of a chirped sinusoid in noise Performance of the LMS algorithm,” IEEE Trans. Signal Processing, vol. 39, pp. 595–602, Mar. 1991.
22. J. R. Treichler , “Transient and convergent behavior of the ALE,” IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-27, pp. 53–63, Feb. 1979.
23. J. R. Zeidler et al., “Adaptive enhancement of multiple sinusoids in uncorrelated noise,” IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-26, pp. 240–254, June 1978.
24. G. R. Elliott and S. D. Stearns, “The Adaptive Line Enhancer applied to chirp detection,” IEEE International Symposium on Circuits and Systems, vol. 2, pp. 1319-1322, 1990.

Sanjeev dhull is working as Asst.Professor in Electronics and Comm. Engg department of Guru Jambheshwar University of Science and Technology ,Hisar.His area of interest is Adaptive signal processing.He had published more than 16 research paper.

Sandeep arya is chairman of Electronics and Comm. Engg department of Guru Jambheshwar University of Science and Technology ,Hisar. His area of interest is Adaptive signal processing and optical signal processing.He had published more than 50 research paper in his field.

O.P.Sahu is working as Associate Professor in National Institute of Technology Kurukshetra. His area of interest is Adaptive signal Processing. Digital electronics. He had published more than 50 research paper in his field.