Available online at www.elixirpublishers.com (Elixir International Journal)

Advanced Engineering Informatics

Elixir Adv. Engg. Info. 39 (2011) 4596-4599

ISSN: 2229-712X

A novel technique to implement LMS adaptive fir filter algorithm using labview

N.J.R. Muniraj

Tejaa Shakthi Innovation Centre, Tejaa Shakthi Institute of Technology, Coimbatore.

ARTICLE INFO

Article history:
Received: 15 July 2011;
Received in revised form:
19 September 2011;
Accepted: 26 September 2011;

Keywords
LMS algorithm,
Active Noise Control,
Adaptive algorithm.

ABSTRACT

In this paper, a novel technique to improve the performance of the LMS method is implemented. As noise is characterized in the low frequency region, the volume of the passive absorbers used increases. Hence, we go in for active methods of suppression. Conventional noise controllers are adaptive filters with the parameters updated mainly on the least mean square basis. It is shown how the least-mean squares algorithm (LMS) can be implemented by using the graphical language LabVIEW. The LMS algorithm under pins the vast majority of current signal processing adaptive algorithm including noise cancellation, beamforming, deconvolution, equalization and so on. It will be shown that labVIEW is a convenient and powerful method of implementing such algorithms.

Introduction

LabVIEW (trademark national instruments) has been in existence since the mid 1980s and is normally associated with virtual instrumentation. Based on the graphical language 'g' it is a block- diagram approach to programming, LABVIEW is a full programming language which is compiled when the 'run' button is pressed. It is perhaps a little harder to learn that approaches such as MATLAB but as will be shown the effort is more than worth the end results.

There is still a wealth of information available on how to use LabVIEW to control a whole range of information available on how to use LabVIEW to control a whole range of instrumentation but far less on the basics of signal processing. perhaps the main exception is the excellent LabVIEW signal processing. The basics of sampling signal generation, filters, matrix manipulation and FFT's are covered there in with numerous activities and real-world applications.

The author was unable to source any information on using LabVIEW for adaptive signal processing using the least-mean-squares(LMS) algorithm.

As the LMS algorithm is the most fundamental of all algorithms used in this area it was decided to write a 'g' program which would perform this function. The type of applications which use LMS are to name a few: system identification, adaptive beam forming (for radar applications), adaptive filtering for hearing aids or generally speech signals, time-delay estimation, active noise cancellation, adaptive equalization for communication channels and some areas of adaptive control.

Active Noise Control

The active method of noise control uses the phenomenon of wave interference. When two coherent waves of the same magnitude and out-of-phase to each, travel in the same direction, they get neutralized owing to destructive interference. This suggests that if a secondary out of phase wave can be constructed and superimposed on the unwanted primary noise, a large reduction in noise level is possible. The active noise control (ANC) system efficiently attenuates low-frequency noise where passive methods are either ineffective or tend to be very expensive or bulky.

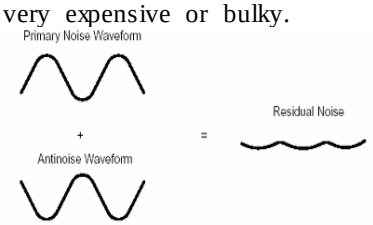

Fig.2.1. Physical concept of active noise control (ANC)

ANC is developing rapidly because it permits improvements in noise control, often with potential benefits in size, weight, volume, and cost.

The creation and superposition of the secondary noise for controlling three- dimensional primary noise is very difficult, since this involves reconstruction of the whole acoustic event. The discussion in this thesis restricts itself to one-dimensional ANC in long ducts. Working with low frequency noise in ducts has got the following advantages: firstly, the sound will travel as plane waves upto a certain frequency called cutoff frequency. The noise of higher frequencies will decay within a short distance from the source. So the mixing of the primary and secondary noise waves is easier. Secondly, the low frequency noise has a longer wavelength, so that the phase angle changes slowly with time. This makes the fine control of phase of secondary wave easier. Hence, a stable interference pattern is possible that results in larger noise reduction. Lastly, the sound wave travels at much slower speed than the electrical signals, so that a large operation time for generating the secondary noise is available, if secondary source is suitably located. The cutoff frequency for a duct depends upon its cross sectional area and the speed of sound inside the duct. For a duct with square cross section, the cutoff frequency is given by

Where a is width of the duct and c is the sound velocity inside the duct. For most of the noise control problems, the fluid inside the duct is air. The sound velocity in air is 343 metres/sec at 30^{o}C. If the cross section of the duct is 130 mm square (used in this work), then the cutoff frequency is 1208 Hz. The control of higher frequency by active method will

Tele:
E-mail addresses: njrmuniraj@yahoo.com

require multiple secondary sources or transverse partitioning of the duct. However, low frequency is the region where conventional silencers are less effective. Thus, active method using wave interference is complementary to the existing silencers for low frequency noise control.

Adaptive Algorithm Used

Adaptive algorithms are used to adjust the co-efficient of digital filter such that the noise is minimized. The algorithm widely used is LMS (Least Mean Square Algorithm).

Some of the classical applications of adaptive filters are system identifications, channel equalization, signal enhancement and signal prediction. The general case of such an application is depicted below.

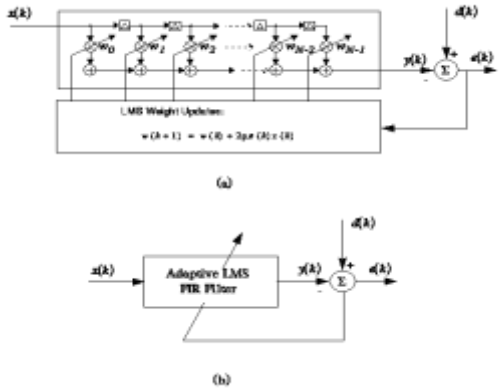

Fig 3.1 fir filter structure

Where the signal x(k) is corrupted by noise n(k), and the signal d(k) is correlated to the noise. When the algorithm converges, the output signal e(k) will be an enhanced version of the signal.

The Mean Square Error (F [e(k)] = [|E [e(k)|^2]) is a quadratic function of the parameters of the weights. This property is important and is used in adaptive filters because it has only one universal minimum value.

This means it is suitable for many types of adaptive algorithms, and will result in a decent convergence behavior. In contrast, IIR filters need more complex algorithms and analysis on this issue.

The LMS algorithm for system identification

Consider the block diagram shown in Figure 1 below. Although the LMS algorithm can be applied to all of the applications mentioned in the previous section, perhaps the easiest to understand and test any algorithm is that of system identification. The basic objective is to find the transfer function of an unknown system. Usually the system is driven by a white-noise source.

The output of the unknown system is labeled the primary input to the algorithm whilst the white-noise source itself is labeled the reference. The coefficients of another filter within the LMS algorithm are then adjusted according to an error (shown on the diagram).

When this error (in fact its average squared value) is at a minimum value, it is considered that the LMS algorithm has converged and the coefficients of the filter within the LMS algorithm then match the unknown system. Probably for the simple example shown it is best thought of as' path balancing' i.e. if the unknown system was placed instead in the reference path then the LMS algorithm would have to converge to the inverse of the unknown system.

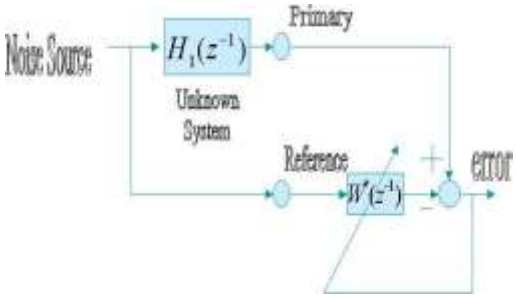

Fig 3.2. LMS algorithm used for system identification

The LMS inputs are described mathematically at time sample 'k' as sk (primary input) and uk(reference input). The unknown system is labeled H1 (z^{-1}) which for convenience is a finite impulse response filter (FIR) of order n with uk as its input and sk as its output. the unknown system need not be FIR but it is easier to check the results.

$H1(z^{-1}) = b0+b1z^{-1}+b^2z^{-2} +\ldots b_nz^{-n}$

Where the coefficients b of the filter are unknown and are to be estimated.

The LMS algorithm is given by

Update the error

$e_k=s_k-X_k^TW_{k-1}$ (1)

update the weight vector estimate W_k

$W_k= W_{k-1}+2\mu X_k e_k$

The weight vector is a column vector

$$W_k= \begin{pmatrix} W_o \\ W1 \\ . \\ . \\ Wn \end{pmatrix}$$

and Xk is the column vector of regressors (or past values of input)given by

$$W_k= \begin{pmatrix} W_o \\ W1 \\ . \\ . \\ Wn \end{pmatrix}$$

N is the system order and there are n+1 weights. The transporse notation (T) in equation (1) therefore makes X a row vector.for the purposes of programming both W and X are arrays or length n. this simple recursion lends itself well to real-time applications and has relatively good tracking ability. The system uk(reference signal) for this input application is assumed to have no dc and have a variance (average power).if a white-noise signal is not available then a signal rich in hormonics can be used (for example a square wave. The stepsize must be chosen carefully. Too large and the algorithm will go unstable,too low and convergence will be slow,it is well known that the condition for convergence . In fact practically often a tenth of the theoretical maximum is used or sometimes one third.for speech signals the variance will vary with time and clearly under such conditions must vary too or be fixed at the worst (smallest) possible case. If the LMS algorithm converges then each weight will in this case correspond to the coefficients of the unknown system.

The LabVIEW LMS virtual instrument

This section will cover the graphical code for implementing the basic LMS algorithm.It is best to start using a simple example and for this an FIR filter of third order (ie four weights) was used driven by white Guassian noise. Figure 4.1 illustrates how this is achieved in the main programming diagram. The LMS algorithm is implemented for convenience as a sub .vi (virtual instrument file).

It is contained within a *While* loop which runs endlessly. The FIR system parameters were chosen arbitrarily as there are no stability issues.

The simplest way to simulate a third order FIR system isto use a formula node. The registers on the main while loop are required to store pastvalues of input (ie regressors).

The input and output to the LMS algorithm are both scalar quantities. As the library function which generates Gaussian white noise is fundamentally an array, it is necessary to make this of length one point and use an *index array* block after this pointing to the zeroth element ie the first point. Otherwise the noise generator output will stay as type array and the connection to the LMS sub will not be possible. In Figure 4.1, sk is the system output and uk the system input with uk1,uk2 and uk3 all past samples of uk.

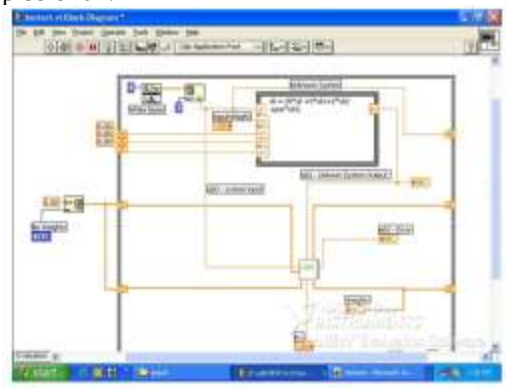

Fig 4.1. The basic LMS virtual instrument

Looking inside the LMS sub itself reveals the diagram shown in Figure 3 below

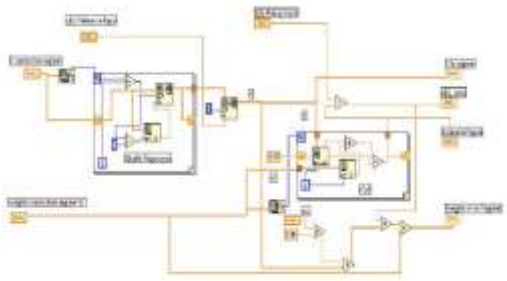

Fig 4.2. The LMS algorithm sub .vi

All inputs are shown to the left and outputs to the right. One of the tasks performed isto shuffle the past samples of the contents in the vector X. For example if we had four weights it would be necessary to perform at each iteration of the main while loop

X[3]=X[2]
X[2]=X[1]
X[1]=X[0]

and finally X[0] = new sample for reference input (u 0).

This is relatively easy to perform in sequential code but in graphical code this is a little more complicated and one solution is shown in Figure 4 below.

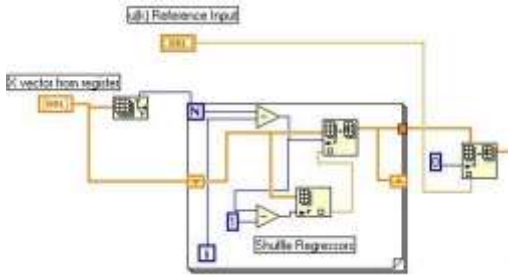

Fig 4.3. Shuffles regressor vector X

The diagram in Figure 4 makes use of the library function *replace array subset* and *index array*. The function *index array* selects the desired element of the X array and the *replace array subset* puts the shuffled value into the array. External to the *for loop*

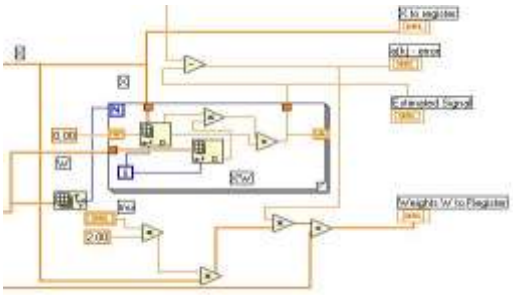

Fig 4.4. Shows how to compute $X^T W$

The arrays W and X are both shown whilst the broken line in the far right is the product . all that is required here is use of the library function index array to select the individual points in the array. They are multiplied and stored in the register. Note that the register is initialized to zero each time the loop is executed. The rest of the LMS algorithm is pretty straight forward as there are only two equation in total. For instance equation 1 only requires the primary input minus the previously to give the scalar. The step length is made up of external input to the LMS sub.vi. in the main panel it is set as a control input. Of course more sophisticated automatic calculation of the step length can be performed but this can easily be added later. The final stage is to implement the vector $W_k = W_{k-1}+2\mu X_k e_k$ has already been calculated. this is done by using a register to store the current values in the Wk and recovering them at the next iteration of the vector W_k . While loop as shown in figure 2, the regressor vector X must also be stored in a register so as not to loose the values there in. the front panel of the LMS virtual instrument is quite basic and is shown in figure 6

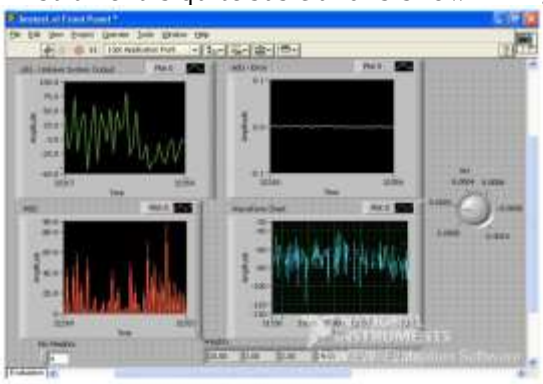

Fig 4.5. Front panel of LMS virtual instrument

Conclusion

The LMS algorithm has been implemented using the graphical programming language ‘g’ (Lab VIEW). It has been

shown how aspects of adaptive signal processing can be quite elegantly implemented using this method and is a powerful alternative to other programming methods. Adaptive array signal processing could be investigated using a similar approach which would give improved results. . In this project, we successfully realized the adaptive noise control, in MATLAB and LABVIEW. These adaptive systems driven by the LMS algorithm have been utilized to achieve an effective noise control,. Furthermore, this project has been explored only a few applications of the LMS algorithm from a vast literature of applications .

References

[1] Johnson,G.W.,LabVIEW *Graphical programming*, McGraw-Hill series,NY,1994

[2] Chugani,M,L,Samant,A.R and Cerna,M., LabVIEW *Signal Processing*, National Instruments Series, Prentice Hall,NJ,1998

[3] S.Haykins ,"Adaptive Filter Theory", Third Edition, Prentice Hall Inc., NJ.1996.

[4] Michael John Sebastian Smith, "Application-Specific Integrated Circuits", Fifth Edition, Pearson Education Inc, Asia, 2001.

[5] Rajeev Murgai,Robert K.Brayton,Alberto Sangiovanni-Vincentelli, "Logic Synthesis for Field Programmable Gate Arrays" ,Kluwer Academic Publishers.

[6]U.Meyer-Baese,"Digital Signal Processing with Field Progrrammable Gate Arrays" Second Edition, Springer International Edition, 2006.

[7] Geoff Bostock,"FPGAs and Programmable LSI:A Designer's Handbook ,1996.

[8] Bernard Widrow, Samuel D.Stearns, "Adaptive Signal Processing", Pearson Education Asia, Second Edition, 2002.

[9] Emmanuel C. Ifeachor, Barrie W. Jervis, "Digital Signal Processing – A. practical approach", Pearson Education Asia, Second Edition, 2002

[10] Treichler, Johnson, Larimore, "Theory and design of Adaptive Filters", Prentice Hall of India Pvt. Ltd., pp 269.

[11] B. Widrow, J.M McCool, M.G.Larimore, and C.R. Johnson Jr., "Stationary and Non stationary Learning Characteristics of the LMS Adaptive Filter", Proc. IEEE, vol.64, pp 1151-1163, August 1976.

[12] C.R. Johnson, Jr. and M.G. Larimore, " Comments on and addition to "An Adaptive Recursive LMS Filter", Proc. IEEE, vol. 65, pp 1399- 1401, September 1977 .

[13] Samir Panitkar, "Verilog HDL", Pearson Education, Second Edition.

[14] J.Basker, "Verilog HDL Primer", BS publication, Second Edition.

Authors Biography

N.J.R. Muniraj is presently working as a Principal of Tejaa Shakthi Institute of Technology, Coimbatore. He has more than 22 years of teaching and five years of industrial experience. He has presented more than 40 National and International papers and published fifteen international journal papers. His research area includes VLSI Signal Processing, Neural Networks, Image Processing and MEMS. He is also heading the Tejaa Shakthi Innovation centre.