LATTICE FILTERS FOR SYSTEM IDENTIFICATION

Randolph L. Moses

Department of Electrical Engineering
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061

Abstract

Several algorithms for recursively updating the exact least squares AR and ARMA parameter estimates have recently been developed. Commonly called lattice algorithms, these methods update optimal coefficient estimates sequentially as new data become available. The algorithms are characterized by computational efficiency, requiring O(p) multiplications and additions for each update. The lattice algorithms also exhibit desirable numerical properties.

This paper applies lattice algorithms to system identification problems. Both the known input and the unknown input cases are considered as are both AR and ARMA models. This paper also discusses order determination, and addresses the problems associated with measurement noise and time varying system parameters.

I. INTRODUCTION

The problem of system identification is that of identifying a model for an unknown system from knowledge of its input and output sequences. (See Figure 1). This problem is generally approached by first specifying a particular structure for the model and then estimating any unknown coefficients in that structure. One such structure is the autoregressive-moving average (ARMA) model, in which the transfer function of the unknown system is represented by a quotient of two polynomials in z^{-1}; that is

$$H(z) = \frac{Y(z)}{U(z)} = \frac{b_0 + b_1 z^{-1} + \ldots + b_q z^{-q}}{1 + a_1 z^{-1} + \ldots + a_p z^{-p}} = \frac{B(z)}{A(z)} \quad . \qquad (1)$$

Here Y(z) and U(z) are the z-transforms of the input and output sequences of the unknown system, and the coefficients $b_0, b_1, \ldots, b_q$ and $a_1, a_2, \ldots, a_p$ are to be estimated.

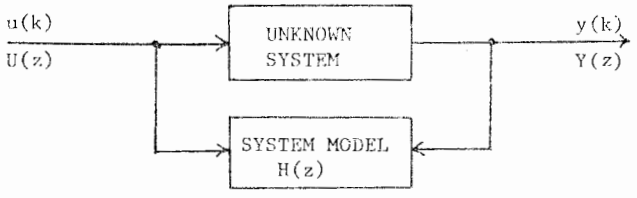

Figure 1. The System Identification Problem.

There are several well-known algorithms for estimating the coefficients of an ARMA model. Of these, the so-called least squares algorithms have gained wide popularity. In least squared algorithms the ARMA coefficients are obtained by minimizing the square of some function associated with the system and model. These algorithms often provide good parameter estimates for a reasonable computational effort.

Most least squares algorithms are block processing algorithms, which operate on a block of data to produce an estimate in a single processing effort. There are, however, many system identification applications in which block processing algorithms are not well suited. Examples include tracking the parameters of a time varying system and adaptive identification for noise cancelling, channel equalization, and system controlling. For many of these applications it is desired to continually update the parameter estimates as new data become available. Algorithms that provide these ongoing updates are called recursive algorithms.

Because recursive algorithms are often used in real time settings, it is important to keep the computational effort required in each time interval to a minimum. Until recently, however, recursive least squares methods were required to perform on the order of p^2 (i.e., $O(p^2)$) multiplications and additions for each update. To decrease the processing effort, approximate least squares algorithms requiring O(p) multiplications and additions per update were developed. However, these approximate methods often provided inferior model estimates and were thus unsuitable in many applications.

Recently, a class of fast recursive least squares algorithms have appeared. These so-called lattice algorithms solve the exact least squares problem at each time step and require only O(p) computations per update. Moreover, they have been shown to exhibit good numerical properties and to permit the capability of tracking slow time variations in the unknown system.

In this paper we present some of the currently available AR and ARMA lattice algorithms. We then propose a new class of ARMA lattice methods, and compare them to other currently available ones. We also outline various modifications to these lattice algorithms that provide better numerical properties and enable tracking of time varying system parameters.

0094-2898/82/0000/0266$00.75 © 1982 IEEE

II. AR MODELING

In autoregressive system identifications we wish to estimate the a_i coefficients in the model

$$H(z) = \frac{1}{1 + a_1 z^{-1} + \ldots + a_p z^{-p}} = \frac{Y(z)}{U(z)} \tag{2}$$

given N measurements y(1), y(2),....,y(N) of the unknown system's output sequence. We shall assume that the input sequence {u(k)} is unmeasurable, zero mean, white noise.

A. The Block Processing Algorithm.

The least squares algorithm for obtaining the AR coefficient estimates is predicated on the inverse transform of equation (2), namely

$$y(k) + \sum_{i=1}^{p} a_i y(k-i) = u(k) \tag{3}$$

We may utilize equation (3) to choose the a_i coefficients to most closely fit the input and output sequences. Since input sequence measurements are not available, we shall replace u(k) in equation (3) by its expected value of zero. The criterion we shall incorporate to determine closeness of fit will be the minimum squared error criterion. Thus, the autoregressive coefficients are chosen to be the minimum squared error solution to the overdetermined system of equations

$$\underline{y} + Y\underline{a} = \underline{0} \tag{4a}$$

where

$$\underline{y} = [y(m), y(m+1), \ldots, y(n)]^T \tag{4b}$$

$$\underline{a} = [a_1, a_2, \ldots, a_p]^T \tag{4c}$$

$$Y = \begin{bmatrix} y(m-1) & y(m-2) & \ldots & y(m-p) \\ y(m) & y(m-1) & \ldots & y(m+1-p) \\ \vdots & & & \\ y(n-1) & y(n-2) & \ldots & y(n-p) \end{bmatrix} \tag{4d}$$

and $\underline{0}$ is the zero vector.

For convenience of notation, we shall assume

$$y(k) = 0 \text{ for } k<1 \text{ and } k>N \tag{5}$$

Equation (4) is the general form for the least squares autoregressive estimation algorithm [8]. Specific versions of this algorithm are realized by choosing particular values for the indeces m and n. Four common choices are:

1) m = p+1 and n = N : the covariance method.
2) m = 1 and n = N : the prewindow method.
3) m = p+1 and n = N+p : the postwindow method.
4) m = 1 and n = N+p : the correlation method.

The solution to equation (4) may also be represented as the system of p equations and p unknowns

$$R\underline{a} = -\underline{r} \tag{6}$$

where $R = Y^T Y$ and $\underline{r} = Y^T \underline{y}$. It is a simple matter to show that the (i,j)th element of R is given by

$$[R]_{i,j} = \sum_{k=m+1}^{n} y(m-i)y(m-j) \tag{7}$$

The jth element of $\underline{r}$ is given by equation (7) with i = 0. The elements of R and $\underline{r}$ are seen to be estimates of autocorrelation lags for {y(k)}. In this context, equation (6) represents an approximation of the well-known Yule Walker equations

$$r_{yy}(k) = \sum_{i=1}^{p} a_i r_{yy}(k-i) = 0 \qquad k \geq 1 \tag{8}$$

where $\{r_{yy}(k)\}$ is the autocorrelation sequence associated with {y(k)}.

The various data windowing methods mentioned earlier correspond to various strategies for estimating the autocorrelation lags $r_{yy}(k)$ in equation (8). These four methods represent a tradeoff between bias in the Yule Walker equation estimates and computational speed. The covariance method assumes no knowledge of the output data outside the measurement interval, but this method is the most computationally burdensome of the four. Conversely, the correlation method implicitly assumes that the output data is zero on both ends of the measurement interval, so there is a resulting bias in the Yule Walker equation estimates. However, the correlation method is computationally the fastest of these four methods. The prewindow and postwindow methods provide a compromise between the two extremes. An analysis of the equation bias and the computational requirements for these four methods can be found in [1].

B. The Fast Recursive Algorithm.

The least squares algorithm presented above involves solving the matrix equation (6). Direct implementation of this algorithm entails calculating the elements of R and $\underline{r}$ from the block of measurements y(1),y(2),....,y(N) and computing the autoregressive coefficient vector as

$$\underline{a} = -R^{-1}\underline{r} \tag{9}$$

This is a block processing implementation, because a single processing effort is performed on a block of data to produce the desired solution.

Recently, a class of fast recursive implementations for least squares AR algorithms has been developed. These recursive implementations provide exact solutions to the least squares algorithm at every time interval by updating previous estimates as each new data measurement y(n) becomes available. They are fast in the sense that only O(p) computations (additions, multiplications and divisions) are needed to update the estimate at each time interval.

The derivation of these fast recursive implementations is somewhat tedious and will not be presented here. The underlying principle in these derivations, though, it to express the estimate at time n as the sum of the estimate at time n-1 (which is known) and a correction term due to the new data point y(n). The algorithm is further refined by expressing the m^{th} order estimate at each time as an orthogonal composition of the m-1st order estimate and another correction term due to the next higher order. By calculating successively

higher orders at each time interval, the p^{th} order parameter estimate may be expressed as the composition of p orthogonal terms. These orthogonal terms can be calculated by performing only $0(p)$ computations, resulting in the fast calculation of the p^{th} order parameter estimate.

As an example of a recursive least square algorithm, let us consider the prewindow version. The initial conditions for the algorithm are:

$$\Delta_{m,0} = r^f_{m,0} = r^b_{m,0} = 0 \qquad m = 0,1,2,\ldots,p-1 \tag{10}$$

As each new data point y(n) becomes available, we have from previous calculations all variables with time index n-1. We first set $f_{0,n} = b_{0,n} = y(n)$. Then, for each order $m = 0,1,2,\ldots,p-1$ we calculate

$$\Delta_{m,n} = \Delta_{m,n-1} + \frac{(f_{m,n})(b_{m,n})}{\gamma_{m,n}} \tag{11a}$$

$$r^f_{m,n} = r^f_{m,n-1} + \frac{(f_{m,n})(f_{m,n})}{\gamma_{m,n}} \tag{11b}$$

$$r^b_{m,n} = r^b_{m,n-1} + \frac{(b_{m,n})(b_{m,n})}{\gamma_{m,n}} \tag{11c}$$

$$\rho^b_{m,n} = \frac{\Delta_{m,n}}{r^f_{m,n}} \quad \text{and} \quad \rho^f_{m,n} = \frac{\Delta_{m,n}}{r^b_{m,n}} \tag{11d}$$

$$f_{m+1,n} = f_{m,n} - (b_{m,n-1})(\rho^f_{m,n}) \tag{11e}$$

$$b_{m+1,n} = b_{m,n-1} - (f_{m,n})(\rho^f_{m,n}) \tag{11f}$$

$$\gamma_{m+1,n} = \gamma_{m,n} - \frac{(b_{m,n-1})(b_{m,n-1})}{r^b_{m,n-1}} \tag{11g}$$

We can see that this algorithm requires 6p additions, 6p multiplications and 6p divisions for each time update.

The last three equations of the recursive prewindow algorithm may be depicted as signals in a filter, as shown in Figure 2.

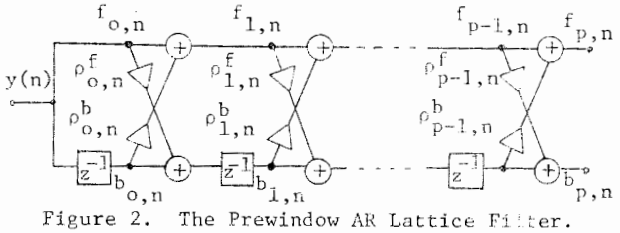

Figure 2. The Prewindow AR Lattice Filter.

Such a filter is termed a lattice filter and its corresponding update procedure is termed a lattice algorithm.

It should be noted that the coefficients of this lattice filter are not the autoregressive coefficients, but rather a set of 2p "reflection" coefficients $\rho^f_{m,n}$ and $\rho^b_{m,n}$. To convert the reflection to the autoregressive coefficients, the following equations are implemented recursively for $m = 1,2,\ldots,p$

$$A_m(z) = A_{m-1}(z) - z^{-1}\rho^f_{m,n}\tilde{A}_{m-1}(z) \tag{12}$$

$$\tilde{A}_m(z) = z^{-1}\rho^b_{m,n}A_{m-1}(z) + \tilde{A}_{m-1}(z)$$

where $A_0(z) = \tilde{A}_0(z) = 1$. The coefficients of $A_p(z)$ are the desired autoregressive parameter estimates from the p^{th} order least squares algorithm.

In order to obtain autoregressive coefficient estimates at each time interval, p^2 additional multiplications and additions are required. However, these extra computations may be done concurrently with the lattice update computations, and for small values of p these extra computations are comparable in number to those required for the lattice update. Moreover, since it is necessary to calculate $A_m(z)$ for $m = 1,2,\ldots,p-1$, the AR parameter estimates for the m^{th} order least squares algorithm are calculated for all lower orders at no additional computational cost. These lower order estimates are often very useful in determining the "best" model order.

III. ARMA MODELING

The task of autoregressive-moving average model identification is to determine the p a_i and q+1 b_j coefficients in the transfer function model

$$H(z) = \frac{Y(z)}{U(z)} = \frac{b_0 + b_1z^{-1} + \ldots + b_qz^{-q}}{1 + a_1z^{-1} + \ldots + a_pz^{-p}} = \frac{B(z)}{A(z)} \tag{13}$$

where {u(k)} and {y(k)} are the input and output sequences, respectively, of the unknown system. Below, we consider two methods for recursively calculating the ARMA model's coefficient estimates.

A. The Two-Dimensional AR Lattice Algorithm

One fast recursive ARMA coefficient estimation procedure is a direct extension of the AR method discussed earlier [6]. In this algorithm the ARMA process is rewritten as a two-dimensional AR process, and a corresponding two-dimensional AR lattice algorithm is used to estimate the desired coefficients.

We can easily reformulate the ARMA model into a two-dimensional AR model by first rewriting equation (13) as

$$A(z)Y(z) = b_0U(z) + B_1(z)U(z) \tag{14}$$

where $B_1(z) = b_1z^{-1} + b_2z^{-2} + \ldots + b_pz^{-p}$.

For this method it is necessary to restrict the numerator and denominator orders to be equal (i.e. p = q). By combining equation (14) with the equation U(z) = U(z) we obtain the desired two-dimensional AR formulation.

$$\begin{bmatrix} A(z) & -B_1(z) \\ 0 & 1 \end{bmatrix}\begin{bmatrix} Y(z) \\ U(z) \end{bmatrix} = \begin{bmatrix} b_0U(z) \\ U(z) \end{bmatrix} \tag{15}$$

The fast recursive algorithm corresponding to this two-dimensional AR formulation is almost identical to the one-dimensional case. The update equations are the same except that scalars are replaced by 2×1 vectors and 2×2 matrices. The complete set of update formulas for the prewindowed version of this ARMA algorithm can be found in [2]

and [6], and the corresponding lattice filter is shown in Figure 3.

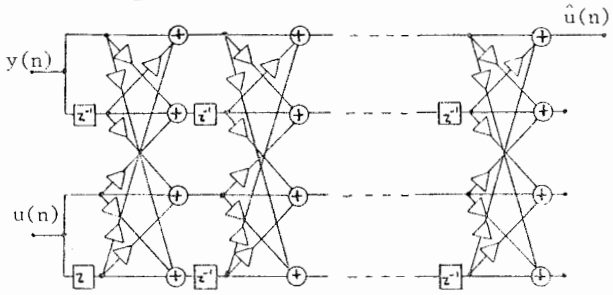

Figure 3. The Prewindow Two-Dimensional AR Lattice Filter.

The number of additions and multiplications required to update the reflection coefficient estimates is still O(p), but it is actually 3 to 4 times as many as needed for the pth order AR lattice since vectors and matrices are being multiplied. However, the increased computational requirement is usually offset by a lower permissable choice of model order p for the ARMA case.

It is important to note that the two-dimensional AR lattice algorithms require measurements of the unknown system's input sequence. If these measurements are not available, they must be estimated. One proposed method for obtaining this estimate is to first set u(k) to zero and then implement the update equations for time k. The lattice filter output is then found to be an estimate of u(k) (see Figure 3). With this estimate of u(k), the lattice equations are implemented a second time to arrive at the final estimate for time k. We can see that this procedure requires two passes of an infinite loop for successively calculating (hopefully) better estimates of u(k). By terminating the iteration after a finite number of passes, a substantial bias on the reflection coefficients may result. The characteristics of this bias have not been studied. Therefore, when the system input measurements are not available, these two-dimensional AR lattice algorithm estimates possess unknown statistical properties.

B. The Fast Recursive High Performance Method.

We propose an alternate approach to recursive ARMA model identification. The proposed method begins by estimating the model's autoregressive coefficients using a lattice algorithm. Once these AR coefficients have been estimated, they are used to effect an efficient method for obtaining the moving average coefficient information. This method does not require explicit knowledge of the unknown system's input sequence. Furthermore, the autoregressive coefficient estimates obtained by this algorithm are consistent and have known bias properties.

The proposed fast recursive ARMA procedure is based on the "high performance" method of autoregressive coefficient estimation [1]. The high performance method provides an effective means of estimating the autoregressive coefficients of an ARMA model given the output measurements y(1),y(2) ....,y(N) and assuming that the input sequence is zero mean, white noise. Basically, an approximation of the model's underlying Yule Walker equations is constructed using the given output measurements. The details of this method's derivation are presented in [1], where it is shown that the desired autoregressive coefficient vector estimate can be determined by solving the system of p equations for $\underline{a}$

$$X^T\underline{y} + X^T Y \underline{a} = \underline{0} \ . \qquad (16a)$$

Here, the vectors $\underline{y}$ and $\underline{a}$ and the matrix Y are given in equation (4) and

$$X = \begin{bmatrix} y(m-1-q) & y(m-q) & \dots & y(m-p-q) \\ y(m-q) & y(m+1-q) & \dots & y(m-p+1-q) \\ \vdots & & & \vdots \\ y(n-1-q) & y(n-2-q) & \dots & y(n-p-q) \end{bmatrix} \qquad (16b)$$

Particular versions of this algorithm are defined by choosing particular values of m and n. Again four commonly chosen versions are the covariance, prewindow, postwindow, and correlation methods, obtained by choosing the same values for m and n as in the autoregressive versions in the previous section.

As a consequence of the similarity between equations (4) and (16), a similar fast recursive implementation for the proposed AR coefficient estimator can be derived. The resulting lattice algorithm for the prewindow method is presented in [1]. The lattice filter corresponding to this recursive algorithm is shown in Figure 4.

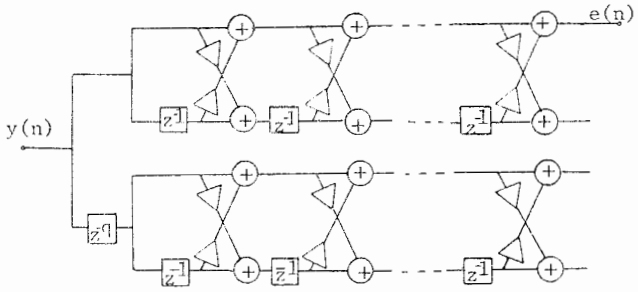

Figure 4. The High Performance Lattice Filter.

It can be seen that this lattice filter is essentially two AR lattices in parallel, with the input of the lower one delayed by q time intervals. The reflection coefficients are calculated using signals from both the top and bottom halves of the lattice, thus providing an interconnection between them. The number of computations required to update this lattice at each time interval is slightly less than double the number required for the same order autoregressive lattice.

Once the autoregressive coefficient estimates have been obtained there are several methods for obtaining estimates of the ARMA model's b_j coefficients. Few of these methods are computationally efficient, however. One method that shows promise is to estimate the q+1 autocorrelation lags

$$\hat{r}_n(i) = \sum_{k=n}^{n-i} e(k+i)e(k) \qquad i = 0,1,2\dots,q \qquad (17)$$

where e(k) is the output signal of the lattice

filter [see Figure 4]. It can be shown that if the lattice filter has correct reflection coefficients, then the $\hat{r}_n(i)$ are unbiased and consistent estimates [within a scalar multiple] of the coefficients of the polynomial $B(z)B(z^{-1})$. Thus, in order to obtain the b_j coefficient estimates from the $\hat{r}_n(k)$ estimates, a (computationally expensive) spectral factorization must be performed. However, in those applications for which explicit knowledge of the b_j coefficients is not needed, this spectral factorization may be avoided, and the $\hat{r}_n(k)$ estimates may be recursively updated in an efficient manner since

$$\hat{r}_n(i) = \hat{r}_{n-1}(i) + e(n)e(n-i) \qquad i = 0,1,\ldots,q \qquad (18)$$

This update requires only q+1 multiplications and q+1 additions at each time step.

Finally, we should note that this proposed lattice provides an effective menas of recursive autoregressive parameter estimation when the output measurements are corrupted by additive noise. To effect this we choose q = p-1 +(the highest nonzero noise autocorrelation lag) to generate the X matrix in equation (16) and we do not perform the moving average calculations.

IV. MODIFICATIONS FOR THE LATTICE ALGORITHM

The scope of the basic lattice algorithms can be extended through the use of two useful modifications - the normalized form and the exponentially weighted form. Both of these modifications are briefly discussed below.

In most lattice algorithms there are some parameters that are monotonically increasing with time. Examples are $r^f_{p,n}$ and $r^b_{p,n}$ in the prewindow AR lattice. This presents a problem, since eventually these parameters will exceed the magnitude limits of the computer implementing the algorithm. To alleviate this problem, normalized versions of the lattice algorithms were developed. The normalized versions eliminate the need to calculate these increasing parameters and often ensure that all other parameters in the algorithm have magnitudes which are less than one, thus permitting fixed point number representation. Moreover, these normalized forms reduce the number of calculations per update, thus reducing the processing requirement. Discussions of normalized forms are found in [2]-[5], and [7].

Another useful modification is the exponential "forgetting factor". This forgetting factor permits the algorithm to weight past estimates by successively smaller amounts, thus enabling the parameter estimates to track slow time variations in the system. The exponential weighting factor is a small modification to most algorithms, often requiring only two or three additional multiplies per time update to implement.

V. CONCLUSIONS

We have presented a class of fast recursive AR and ARMA least squares identification algorithms which provide a computationally fast means of recursively estimating the coefficients of an AR or ARMA model. The AR lattice algorithms have been extensively studied, and as a result their numerical and statistical properties are well understood. ARMA lattice algorithms, on the other hand, are newer, and their characteristics are not as well understood.

After presenting an overview of least squares estimation procedures and AR lattice algorithms, we discussed two classes of ARMA lattice methods. The first converts an ARMA realization problem to a two-dimensional AR problem. When the unknown system's input measurements are available, the statistical characteristics of the resulting parameter estimates are known. When no input measurements are available, however, this algorithm must form an estimate of the input, and the resulting procedure has an unknown statistical characterization. The ARMA lattice algorithm we proposed, on the other hand, does not require input measurements and still provides known statistical properties for the autoregressive coefficients. The major drawback with the proposed ARMA method is that the model's b_j coefficient estimates cannot be explicitly obtained in a computationally efficient manner. It seems, then, that no one recursive ARMA estimation procedure is best for all applications.

Since both of the ARMA lattice algorithms are fairly new, little work has been performed on simulation experiments for each method, and almost no work has been aimed toward experimentally comparing the two methods. Furthermore, a few drawbacks present in these algorithms may still be eliminated. These areas would prove a useful course of future study.

REFERENCES

[1] J. A. Cadzow and R. L. Moses, "An Adaptive ARMA Spectral Estimator", Proc. Spectral Estimation Workshop, August 17-18, 1981, Hamilton, Ontario.

[2] B. Friedlander and S. Mitra, "Speech Deconvolution by Recursive ARMA Lattice Filters", Proc. 1981 Int. Conf. Acoustics, Speech and Signal Processing, Atlanta, GA, pp.343-6.

[3] D. Lee and M. Morf, "Recursive Square Root Ladder Estimation Algorithms", Proc.1980 IEEE Conf. Acoust., Speech, and Sig. Proc.,Denver, Co., pp. 1005-1017.

[4] B. Friedlander, "Recursive Algorithms for Ladder Forms", SCI Technical Memorandum, 1980.

[5] B. Friedlander, "Recursive Algorithms for Pole-Zero Ladder Forms", SCI Tech. Memorandum, 1980.

[6] J. Makhoul, "Stable and Efficient Lattice Methods for Linear Prediction, IEEE Trans. Acoust., Speech, Signal Proc., Vol. ASSP-25, No.5, October 1977, pp.423-428.

[7] B. Porat, B. Friedlander and M. Morf, "Square Root Covariance Ladder Algorithms, Proc. 1981 IEEE Conf. Acoust. Speech, Sig. Proc. Atlanta, GA, pp. 877-880.

[8] M. Morf, B.Dickenson, T.Kailath and A. Vieira, "Efficient Solution of Covariance Equations for Linear Prediction, IEEE Trans. Acoust. Speech, & Sig. Proc., Vol.ASSP-25, Oct. 1977, pp429-433.

[9] K. Ogino, "Computationally Fast Algorithms for ARMA Spectral Estimation", Ph.D. Dissertation, Virginia Poly. Inst. & State Univ., June 1981.