REPRINT
REMOVED

22 P.

Semi-Annual Report

to

NASA-Ames Research Center
Grant Award No. NAG 2-304

NASA Technical Officer: Karl Anderson

for

A Survey of the State of the Art and Focused Research in Range Systems - Task II

Principal Investigator
Kung Yao

Electrical Engineering Department
University of California
Los Angeles, California 90024

June 1986

(NASA-CR-176988) A SURVEY OF THE STATE OF THE ART AND FOCUSED RESEARCH IN RANGE SYSTEMS, TASK 2 Semiannual Report (California Univ.) 22 p CSCL 09C

N86-30069
THRU
N86-30072
Unclas
G3/33 43497

In the period from January to June 1986, the following research activities and publications performed under NASA-Ames Research Contract Grant Award Number NAG-2-304 are enclosed:

1. C. Y. Chang and K. Yao, "On Some Equivalent Configurations of Systolic Arrays," Proc. of the Twentieth Ann. Conf. on Information Sciences and Systems, Princeton, NJ, March 1986.

2. S. Kalson and K. Yao, "Results in Least-Square Estimation Algorithms with Systolic Array Architectures," in Digital Communications, edited by E. Biglieri and G. Prati, Elsevier Science Press, 1986, pp. 235-249.

3. K. Konstantinides and K. Yao, "Modeling and Equalization of Nonlinear Bandlimited Satellite Channels," Conf. Record of Inter. Conf. on Communications, June 1986, pp. 1622-1626.

4. M. J. Chen and K. Yao, "On Realizations of Least-Squares Estimation and Kalman Filtering by Systolic Arrays," Proc. of Inter. Workshop on Systolic Array, Oxford, England, June 1986.

ON SOME EQUIVALENT CONFIGURATIONS
OF SYSTOLIC ARRAYS

CD146017 D1-58.

C.Y. Chang and K. Yao
Electrical Engineering Dept.
University of California
Los Angeles, CA 90024

N86-30070

ABSTRACT

A systematic approach is presented for designing systolic arrays and their equivalent configurations for certain general classes of recursively formulated algorithms. A new method is also introduced to reduce the input bandwidth and storage requirements of the systolic arrays through the study of dependence among the input data. Many well known systolic arrays can be rederived and also many new systolic arrays can be discovered by this approach.

I. INTRODUCTION

A systolic array is a network of processors that rhythmically process and pass data among themselves. It provides pipelining, parallelism, and simple adjacent neighbor cell interconnection structure so that it is suitable for VLSI implementation. While most of the earlier systolic array algorithms were discovered heuristically [1-3], there has been various work on systematic approaches to the design of systolic array algorithms [4-6]. In this paper, we shall present a systematic approach for designing systolic arrays and especially focus on their equivalent configurations for certain general classes of recursively formulated algorithms. In order to reduce the input bandwidth and storage requirements of the systolic arrays, the dependence among the input data is also investigated in details. It is shown that many well known systolic arrays can be rederived and also many new systolic arrays can be discovered by this systematic approach. For simplicity of illustration, we mainly consider the linear systolic array in this paper. The same idea can also be generalized to the two dimensional mesh-connected systolic arrays.

II. IMPLEMENTATION OF RECURSIVELY FORMULATED ALGORITHMS

Consider two simple but important ways of data flow pattern in a linear systolic array as shown in Figure 1 and 2. In these two figures, P_i, Q_j, and b_{ij} are three given input data sequences and R_j is to be the output data sequence, where $0 \leq i \leq m-1$ and $0 \leq j \leq n-1$. For the systolic array shown in Figure 1, Q_j and R_j are stored in the j^{th} processor, where R_j will be updated while P_i is moving to the right and b_{ij} is moving down. For the systolic array shown in Figure 2, P_i is stored in the i^{th} processor and R_j will be updated as it is moving to the right with Q_j while b_{ij} is moving down. All of the data movements are synchronized. The R_j's will successively have the required output data after m steps. For convenience, according to the R_j's behavior of these two systolic arrays, they are respectively named as R-stay and R-move linear systolic arrays. There is great similarity between these two systolic arrays. It can be shown that a large class of interesting problems in the real world can be implemented by these two types of linear systolic arrays. Besides, various different but equivalent configurations of linear systolic arrays can also be derived from them.

Procedure 1 : Given any problem which can be formulated so that it has P_i, Q_j, and b_{ij} as three input data sequences and R_j as the output data sequence, where $0 \leq i \leq m-1$ and $0 \leq j \leq n-1$, if R_j can be generated through the following recurrence equation

$$R_j^{[i+1]} = f(P_i, Q_j, b_{ij}; R_j^{[i]}), \qquad (1)$$

where $R_j^{[0]}$ contains some initial value, f is any function of four variables P_i, Q_j, b_{ij}, and $R_j^{[i]}$, and $R_j^{[m]}$ is the required output data R_j, then this problem can be implemented by the R-stay linear systolic array of n processors and the R-move linear systolic array of m processors. □

The complexity and the configuration of the systolic array depend on the complexity of the function f and the generation procedure of b_{ij}. Some regularity and dependence among b_{ij}'s may greatly simplify the whole system.

III. MAPPING INTO FAN-IN TYPE LINEAR SYSTOLIC ARRAY

Note that for the two linear systolic arrays shown in Figure 1 and 2, the input bandwidth and storage requirements are large in comparison to the number of processors in the array, which may be either infeasible or inefficient for many applications of interests. This is mainly because the dependence among the b_{ij}'s is not efficiently utilized so that each processor needs its own external input connection due to the existence of all the b_{ij}'s. It is expected that under certain circumstances not all of these external input connections are required. In this paper, we are also very interested in the issue of reducing the input bandwidth and storage requirements by showing under what conditions these external input connections can be removed so that only the very first processor is allowed to have such a connection, i.e., the input sequences can only be fanned in through the systolic array. It is shown that the existence of certain patterns of dependence among the b_{ij}'s allows themselves to be fanned-in generated by slightly modifying the operations involved in each processor without losing the property of adjacent neighbor interconnection structure. These conditions are shown in the following two procedures.

Procedure 2 : For the R-stay linear systolic array, if b_{ij} can be determined through the following dependence equation

$$b_{ij} = T(b_{i-1,j}; b_{i,j-1}; u_i; v_j), \qquad (2)$$

where u_i is a variable which depends only on i, v_j is a variable which depends only on j, and T is a function of four variables, then b_{ij} can be generated by the fan-in scheme systolic array as shown in Figure 3 rather than being broadcast as shown in Figure 1. Also note that $b_{-1,j}$ as well as v_j, which depends only on j, can be preloaded in the j^{th} processor, and $b_{i,-1}$ as well as u_i, which depends only on i can be used as a fanned-in input sequence. □

Note that for the R-stay linear systolic array shown in Figure 1, if b_{ij} is the current input to the j^{th} processor, then $b_{i-1,j}$ is the previous input to the j^{th} processor and $b_{i,j-1}$ is the previous input to the $(j-1)^{st}$ processor. It is understandable that in order to avoid the violation of the adjacent neighbor interconnection structure, b_{ij} can only depend on $b_{i-1,j}$ and $b_{i,j-1}$ as well as the data that can be preloaded and the data that can be fanned in, which is what Procedure 2 is about. In general, the systolic array shown in Figure 3 has two sets of input data. One of them consists of three fanned-in data sequences, P_i, u_i, and $b_{i,-1}$, which depend only on the i index, and the other set consists of three preloaded data sequences, Q_j, v_j and $b_{-1,j}$, which depend only on the j index, where u_i, v_j, $b_{i,-1}$ and $b_{-1,j}$ are used to generate all the b_{ij}'s. For each processor, four registers are required, namely Q_p, V_p, B and R, where registers Q_p and V_p are used to store the preloaded data Q_j and v_j respectively. Initially register B is loaded as $b_{-1,j}$ and register R is set to be $R_i^{[0]}$, both of which will be updated as the systolic array start operation. The reason to include so many data sequences is to take care of the general cases. However, it is expected that in many applications, not all of these fanned-in and preloaded data sequences are required. It is often the case that the fan-in generation process of b_{ij} simply depends on two or three data sequences which can either be fanned-in or preloaded. Similarly for the R-move linear systolic array, very similar results can be obtained as follows.

Procedure 3 : For the R-move linear systolic array, if b_{ij} can be determined through the following dependence equation

$$b_{ij} = T(b_{i-1,j}; b_{i,j-1}; u_i; v_j), \tag{3}$$

where u_i is a variable which depends only on i, v_j is a variable which depends only on j, and T is a function of four variables, then b_{ij} can be generated by the fan-in scheme systolic array as shown in Figure 4 rather than being broadcast as shown in Figure 2. Also note that $b_{i,-1}$ as well as u_i, which depends only on i, can be preloaded in the i^{th} processor, and $b_{-1,j}$ as well as v_j, which depends only on j, can be used as a fanned-in input sequence. □

Note that for the R-move linear systolic array shown in Figure 2, if b_{ij} is the current input to the i^{th} processor, then $b_{i,j-1}$ is the previous input to the i^{th} processor and $b_{i-1,j}$ is the previous input to the $(i-1)^{st}$ processor. What procedure 3 says simply repeats the fact that in order to avoid the violation of adjacent neighbor interconnection structure, b_{ij} can only depend on $b_{i-1,j}$ and $b_{i,j-1}$ as well as the data that can be preloaded and the data that can be fanned in. In general, the systolic array shown in Figure 3 has two sets of input data. One of them consists of three fanned-in data sequences, Q_j, v_j, and $b_{-1,j}$, which depend only on the j index, and the other set consists of three preloaded data sequences, P_i, u_i, and $b_{i,-1}$, which depend only on the i index, where u_i, v_j, $b_{i,-1}$ and $b_{-1,j}$ are used to generate all the b_{ij}'s. For each processor, three registers are required, namely U_p, B and P, where registers P and U_p are used to store the preloaded data P_i and u_i. Initially register B is loaded as $b_{i,-1}$ and output data R_j is set to be $R_j^{[0]}$, both of which will be updated as the systolic array start operation.

The previous three procedures provide a rather systematic approach to design the systolic array architecture for the implementation of a given problem. At first, by checking the existence of the recurrence relationship as shown in equation (1), we are able to know if there exist any systolic arrays as shown in Figure 1 and 2. Next, by checking the dependence among the b_{ij}'s as shown in equations (2) and (3), we are able to know the existence of the fan-in type systolic arrays as shown in Figure 3 and 4 so that only small input bandwidth and storage are required. The key issue is in how to search for the recurrence function f and the dependence function T. It is expected that there may exist several different forms of functions due to different possible approaches to formulate a given problem. Various forms of these functions simply create many different but equivalent configurations of systolic arrays. Also note that in the previous discussion, P, Q, b, u, and v are somewhat treated as single variables, however it is clear that they can be set of variables and the same results still hold. This approach can be applied to design systolic arrays for many interesting problems in the real world. Various new configurations of systolic arrays can be derived. In the next section, we shall illustrate this design approach by considering the DFT algorithm.

IV. SYSTOLIC ARRAY ARCHITECTURE FOR DISCRETE FOURIER TRANSFORM

Given n discrete data a_i in the time domain, where $0 \le i \le n-1$, and n discrete frequencies $W_j = (e^{i2\pi/n})^j$ in the frequency domain, where $0 \le j \le n-1$, the discrete Fourier transform (DFT) is to compute

$$y_j = a_{n-1}W_j^{n-1} + a_{n-2}W_j^{n-2} + \dots + a_1W_j + a_0.$$

Let

$$f(P, Q, b; R) = (R \times b) + P.$$

By induction, it can be shown that by letting

$$y_j^{[i+1]} = (y_j^{[i]} \times W_j) + a_{n-i-2} \tag{4}$$

and $y_j^{[0]} = a_{n-1}$, then $y_j^{[n-1]} = y_j$, is the required output. The existence of a recurrence function f and the satisfaction of the recurrence relationship guarantee that there exists systolic arrays for the implementation of discrete Fourier transform as shown in Figure 5 and 6.

It can be seen from Figure 5 and 6 that the b_{ij}'s are not totally independent. Note that $P_i = a_{n-i-2}$ and $b_{ij} = W_j$. In order to see if b_{ij} can be fanned-in generated, let us examine the data dependence among the b_{ij}'s. Many different forms of dependence function T exist. For example,

$$\begin{aligned} b_{ij} &= T(b_{i-1,j};\ b_{i,j-1};\ u_i;\ v_j) \\ &= v_j, \end{aligned} \tag{5}$$

where $v_j = W_j$. The pair of systolic arrays based on equations (4) and (5) are shown in Figure 7 and 8. The systolic array shown in Figure 8 is the well known systolic DFT [2], whose discovery appears to be heuristic rather than in a systematic manner as from our approach. For another example of T function, note that

$$\begin{aligned} b_{ij} &= W_j = W_1^{\ j} = W_1^{\ j-1}W_1 \\ &= b^j_{i,j-1}W_1, \end{aligned}$$

i.e.,

$$\begin{aligned} b_{ij} &= T(b_{i-1,j};\ b_{i,j-1};\ u_i;\ v_j) \\ &= b_{i,j-1}u_i, \end{aligned} \tag{6}$$

where $u_i = W_1$ and $b_{i,-1} = W_1^{-1}$, which can be either used as fanned-in sequences of the R-stay linear systolic array or preloaded in the i^{th} processor of the R-move linear systolic array. The pair of systolic arrays based on equations (4) and (6) are shown in Figure 9 and 10.

Another interesting issue is that the type of function f used in this example does not belong to the class of general matrix vector multiplication. This confirm the fact that the class of problems covered in the Procedure 1 really contains not only the class of general matrix vector multiplication. As well known, there are two different ways to consider the discrete Fourier transform. One shows that the DFT is a special case of the evaluation of a polynomial and the other shows that the DFT is a special case of general matrix vector multiplication. The first way was just considered in this example. Let us see what can be obtained by following the second way. Let

$$f(P,\ Q,\ b;\ R) = R + (P \times b).$$

By induction, it can be shown that by letting

$$y_j^{[i+1]} = y_j^{[i]} + (a_i \times W_j^{\ i}), \tag{7}$$

and $y_j^{[0]} = 0$, then $y_j^{[n]} = y_j$, is the required output. The existence of a new recurrence function f and the satisfaction of the recurrence relationship guarantee that there exists systolic arrays for the implementation of DFT as shown in Figure 11 and 12.

From Figure 11 and 12 it can also be seen that the b_{ij}'s are not totally independent. Note that $P_i = a_i^j$ and $b_{ij} = W_i^{\ i}$. Let us examine the data dependence among the b_{ij}'s. Note that

$$\begin{aligned} b_{ij} &= W_i^{\ i} = W_i^{\ j} = W_i^{\ j-1}W_i = W_{j-1}^{\ \ i}W_i \\ &= b^j_{i,j-1}W^i_i, \end{aligned}$$

i.e.,

$$\begin{aligned} b_{ij} &= T(b_{i-1,j};\ b_{i,j-1};\ u_i;\ v_j) \\ &= b_{i,j-1}u_i, \end{aligned} \tag{8}$$

where $u_i = W_i$ and $b_{i,-1} = W_i^{-1}$, which can be either used as fanned-in sequences of the R-stay linear systolic array or preloaded in the i^{th} processor of the R-move linear systolic array. The pair of systolic arrays based on equations (7) and (8) are shown in Figure 13 and 14. Also note that

$$\begin{aligned} b_{ij} &= W_j^{\ i} = W_j^{\ i-1}W_j \\ &= b^j_{i-1,j}W^j_j, \end{aligned}$$

i.e.

$$\begin{aligned} b_{ij} &= T(b_{i-1,j};\ b_{i,j-1};\ u_i;\ v_j) \\ &= b_{i-1,j}v_j, \end{aligned} \tag{9}$$

where $v_j = W_j$ and $b_{-1,j} = W_j^{-1}$, which can be either preloaded in the j^{th} processor of the R-stay linear systolic array or used as fanned-in sequences of the R-move linear systolic array. The pair of systolic arrays based on equations (7) and (9) are shown in Figure 15 and 16.

This DFT example shows that under certain circumstances it is possible to formulate a given problem in several different ways to implement with various different but equivalent configurations of systolic arrays.

V. CONCLUDING REMARKS

A systematic approach is presented for designing systolic arrays and deriving their equivalent configurations for certain general classes of recursively formulated algorithms. This approach can be considered as a two-stage design procedure. In the first stage, the existence of recursiveness is investigated. If it exists, according to the same formulation the input data are classified into three parts, two of them, P_i and Q_j, depend only on one index, and another one of them, namely b_{ij} depends on both index i and j, so that the systolic arrays shown in Figure 1 and 2 apply. However, for certain applications, it is either infeasible or inefficient to store all of the b_{ij}'s. In the second stage, the dependence among the b_{ij}'s is then investigated to see if it can be used to fan-in generate the b_{ij}'s through the data sequence that can either be preloaded or fanned in. For a given problem, various formulations of the recursive property and the dependence among the b_{ij}'s are possible, which simply lead to many different but equivalent configurations of systolic arrays.

So far we mainly deal with the linear systolic arrays. However, the same technique can be easily generalized to the two dimensional mesh-connected systolic arrays, since the mesh-connected systolic arrays can be simply treated as the concatenation of many linear systolic arrays.

VI. ACKNOWLEDGEMENT

This work was partially supported by the NASA/Ames research contract NAG-2-304.

VII. REFERENCES

1. H. T. Kung and C. E. Leiserson, 'Systolic Arrays (for VLSI),' Proc. Symp. Sparse Matrix Computations and Their Applications, Nov. 2-3, 1978, pp. 256-282.
2. H. T. Kung, 'Why Systolic Architectures,' Computer, Jan. 1982, pp. 37-45.
3. H. T. Kung, 'Let's design algorithms for VLSI systems,' in Proc. Caltech Conf. on VLSI, pp. 65-90, Jan. 1979.
4. P. R. Cappello and K. Steiglitz, 'Unifying VLSI Array Design with Geometric Transformations,' Proc. Int. Conf. on Parallel Processing, pp. 448-457, Bellaire, Michigan, Aug. 1983.

5. D. I. Moldovan, 'On the Design of Algorithms for VLSI Systolic Arrays,' Proc. IEEE, V 71, N 1, pp. 113-120, Jan 1983.
6. W. L. Miranker and A. Winkler, 'Spacetime Representations of Computational Structures,' Computing, V 32, 1984.

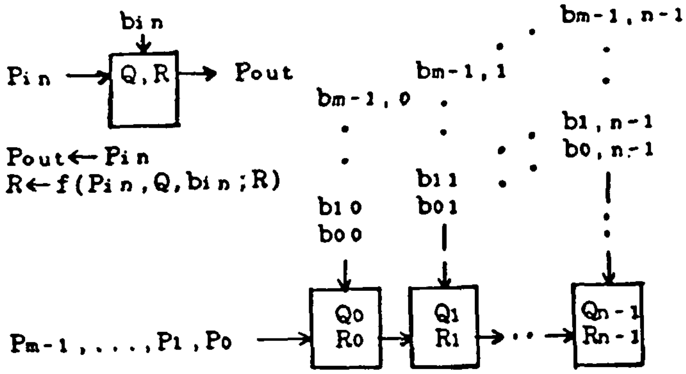

Figure 1: The R-stay linear systolic array.

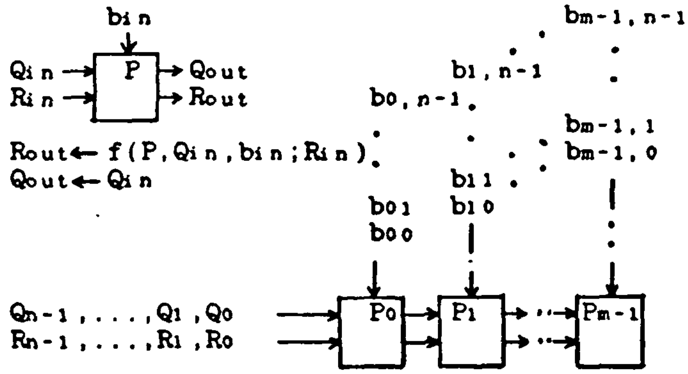

Figure 2: The R-move linear systolic array.

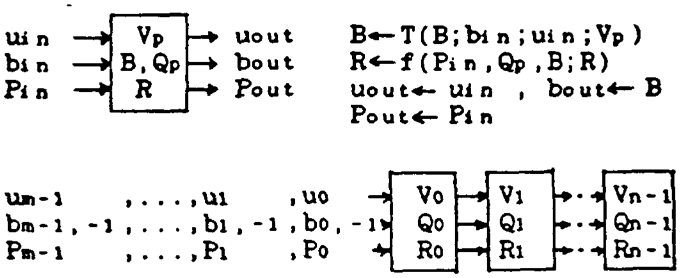

Figure 3: The fan-in scheme of R-stay linear systolic array. Note that the register B in the j^{th} processor is initially loaded with $b_{-1,j}$.

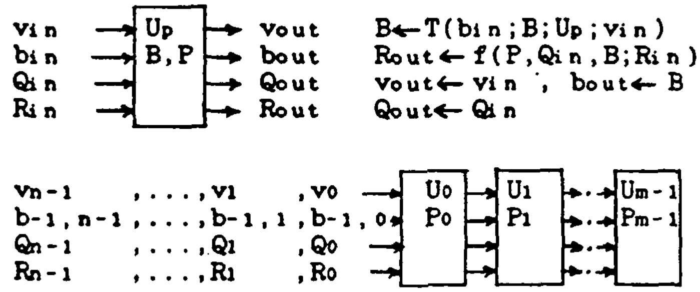

Figure 4: The fan-in scheme of R-move linear systolic array. Note that the register B in the i^{th} processor is initially loaded with $b_{i,-1}$.

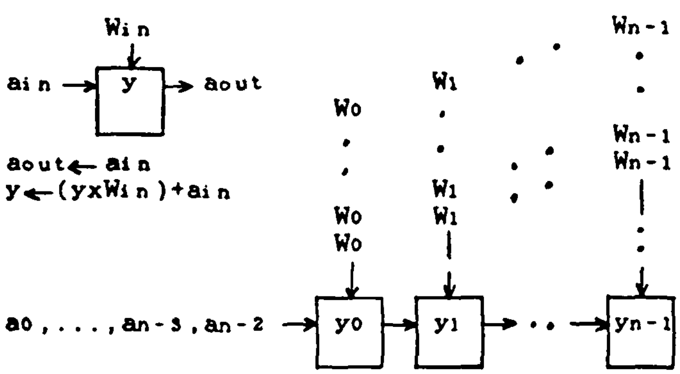

Figure 5: R-stay linear systolic array of discrete Fourier transform based on equation (4).

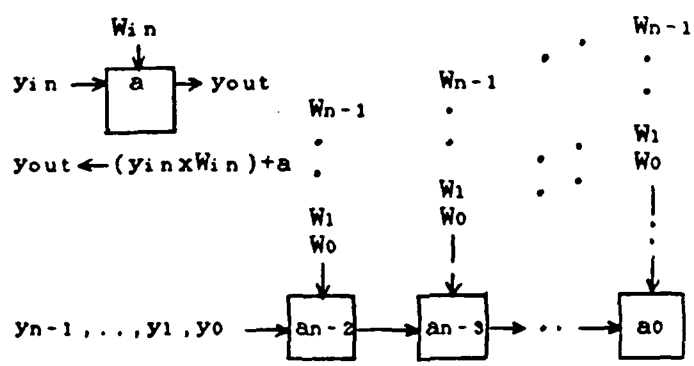

Figure 6: R-move linear systolic array of discrete Fourier transform based on equation (4).

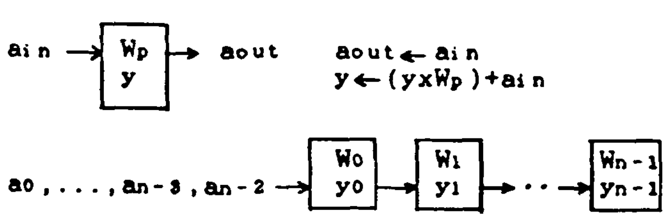

Figure 7: R-stay linear systolic array of discrete Fourier transform based on equations (4) and (5).

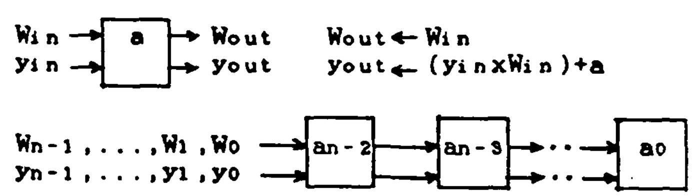

Figure 8: R-move linear systolic array of discrete Fourier transform based on equations (4) and (5)

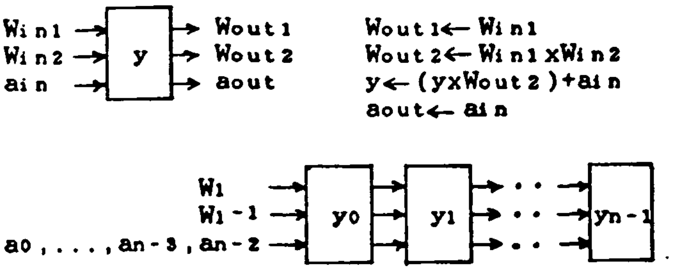

Figure 9: R-stay linear systolic array of discrete Fourier transform based on equations (4) and (6).

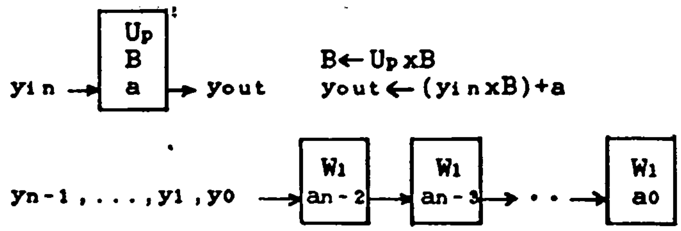

Figure 10: R-move linear systolic array of discrete Fourier transform based on equations (4) and (6). Note that register U_p is preloaded with W_1 and register B is initially loaded with W_1^{-1}.

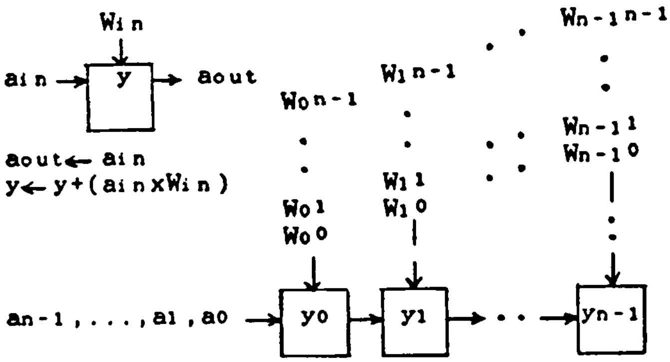

Figure 11: R-stay linear systolic array of discrete Fourier transform based on equation (7).

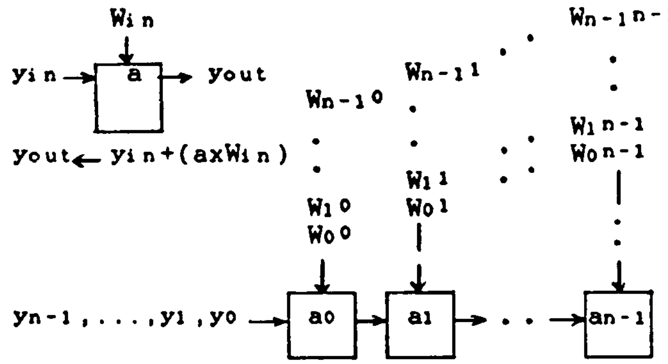

Figure 12 : R-move linear systolic array of discrete Fourier transform based on equation (7).

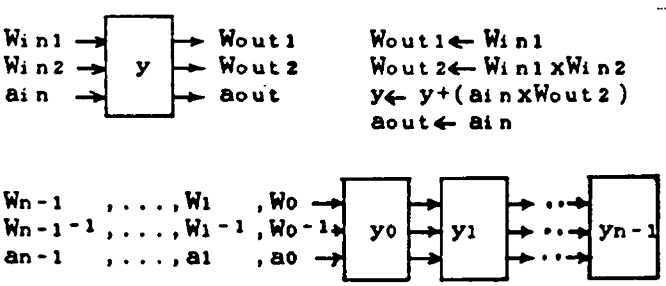

Figure 13: R-stay linear systolic array of discrete Fourier transform based on equation (7) and (8).

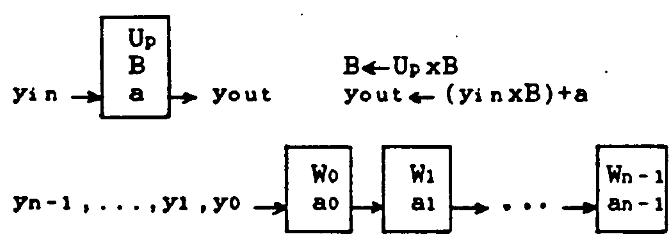

Figure 14: R-move linear systolic array of discrete Fourier transform based on equations (7) and (8). Note that in the i^{th} processor, register U_p is preloaded with W_i and register B is initially loaded with W_i^{-1}.

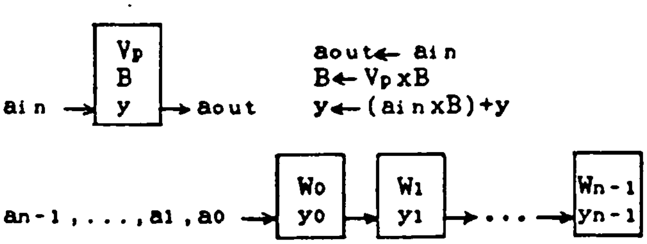

Figure 15: R-stay linear systolic array of discrete Fourier transform based on equations (7) and (9). Note that in the j^{th} processor, register V_p is preloaded with W_j and register B is initially loaded with W_j^{-1}.

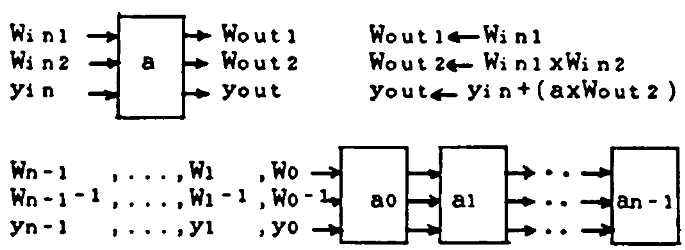

Figure 16: R-move linear systolic array of discrete Fourier transform based on equations (7) and (9).

N86 - 30071 D2 5P.

18334

MODELING AND EQUALIZATION OF NONLINEAR BANDLIMITED SATELLITE CHANNELS

HR825980 CD146017

K.Konstantinides
Hewlett-Packard Laboratories
Palo Alto, CA 94304

K.Yao
Electrical Engineering Department
University of California
Los Angeles, CA 90024

ABSTRACT

In this paper we consider the problem of modeling and equalization of a nonlinear satellite channel. The channel is assumed to be bandlimited and exhibits both amplitude and phase nonlinearities. A discrete time satellite link is modeled under both uplink and downlink white Gaussian noise. Under conditions of practical interest, *a simple and computationally efficient* design technique for the minimum mean square error linear equalizer is presented. The bit error probability and some numerical results for a BPSK system demonstrate that the proposed equalization technique outperforms standard linear receiver structures.

I. INTRODUCTION

The problem of nonlinear channel modeling and equalization is of analytical and practical interest. An important example of a nonlinear channel is a digital satellite communication link, which uses a Traveling Wave Tube (TWT) amplifier operating in a near saturation region. The TWT exhibits nonlinear distortion in both amplitude (AM/AM conversion) and phase (AM/PM conversion). In addition, at high transmission rates the channel's finite bandwidth causes a form of distortion known as Intersymbol Interference (ISI).

In this paper, we will examine the problem of modeling and equalizing this type of nonlinear satellite communication link. The observed data are corrupted by additive white noise, uncorrelated with the input data.

A number of other researchers have studied this problem. Fredricsson [1]. considered a QPSK system and specified an optimum linear receiver filter using a Mean Square Error (MSE) criterion. The channel nonlinearity in [1] was handled via successive number of linearizations Mesiya et al. [2-3] analyzed the BPSK system. In [2] a maximum likelihood receiver was considered, while in [3] a simpler linear receiver, based on the MSE criterion, was presented. In both [2] and [3], the nonlinearity of the TWT is expressed in terms of Bessel function integrals. The MSE criterion was also applied by Biglieri et al. [4] in their derivation of an optimum linear receiver. In [1], [3], and [4], the authors work in the frequency domain, and the solution is given in terms of integral equations that usually have to be solved using numerical techniques.

In [5], Ekanayake and Taylor presented a suboptimum maximum-likelihood type decision feedback receiver. However, because of the analytical complexity of their solution, they approximate the TWT with a hard limiter. A different modeling approach was taken by Benedetto et al. [6]. First, they identify the whole channel using a Volterra Series expansion [7]. Then they suggest a nonlinear equalizer, based again on the MSE criterion. Although at the output of a nonlinear equalizer the MSE is smaller than the MSE at the output of a linear equalizer, it is not completely clear if there is a significant improvement in the probability of error performance of the system to justify the complexity of the nonlinear receiver.

In this paper, we present the design and performance analysis of the optimum linear MSE receiver for a nonlinear satellite channel. While the methods considered here are applicable to various in-phase and quadri-phase modulation systems, for simplicity and lack of space we will illustrate this approach by using only BPSK examples. More generalized results will be presented elsewhere. There are two major differences between our design as compared to the above reviewed approaches. First, we use a very simple model for the input-output relationship of the TWT amplifier, proposed first by Saleh [8]. Second, by working in the discrete time domain we avoid the complex integral equations of the other approaches. In addition, a fast and simple iterative algorithm [9] permits the easy computation of the autocorrelation coefficients of the output of the nonlinear system. Thus, we are able to obtain a new simple and computationally efficient linear equalization technique under the MSE criterion. Based on the same modeling approach, a zero forcing type of linear equalizer was also presented in [10].

In Section 2, a simplified model for a typical satellite link is presented and the corresponding BPSK discrete model is derived. The optimum MSE equalizer is presented in Section 3. In Section 4, the probability of error performance of the receiver is derived. Finally in Section 5 some numerical examples, and comparisons with standard linear receivers are presented.

II. CHANNEL MODELING

Consider the simplified model of a digital satellite communication channel as shown in Figure 1. We will examine each one of the different subsystems composing this model. This study will enable us to derive an equivalent discrete model. By working in discrete time we will avoid the analytical problems arising with continuous signals. Our analysis is similar to that of Ekanayake and Taylor [5].

The source output is a random sequence $\{U(n)\}$ of equally probable uncorrelated symbols. Thus, in a BPSK system, $U(n)=\{1,-1\}$ at $n=0, T, 2T, \ldots$, where $P[U(n) = 1] = P[U(n) = -1] = 0.5$, $E[U(n)U(n-k)]=0$ for $k \neq 0$, and T^{-1} is the signaling rate.

Let $p(t)$ denote a pulse shaping function. Often it can be a rectangular function of unit amplitude over a time period of length T. In any case, the output of the modulator can be expressed in the form

CH2314-3/86/0000-1622 $1.00 © 1986 IEEE

$$s(t) = \sum_{n=-\infty}^{\infty} U(n)p(t-nT)\cos\omega_c t, \tag{1}$$

where ω_c is the carrier frequency.

We shall assume that the transmission filter is the one which determines the channel bandwidth. This filter is also responsible for the creation of ISI. Let $G(t) = 2g(t)\cos\omega_c t$ be the impulse response of this filter, where g(t) is the impulse response of a corresponding low pass filter. Then the output of this filter can be expressed as

$$s_u(t) = \sum_{n=-\infty}^{\infty} U(n)h(t-nT)\cos\omega_c t, \tag{2}$$

where $h(t) = g(t)^* p(t)$. The purpose of our analysis is the design of a receiver structure for the estimation of the transmitted source symbol during the n th signaling interval $nT \le t \le (n+1)T$. Thus during the n th signaling interval (2) can be rewritten as

$$s_u(t) = U(n)h(t-nT)\cos\omega_c t + \sum_{i \ne n} U(i)h(t-iT)\cos\omega_c t, \tag{3}$$

$$nT \le t \le (n+1)T.$$

The first term in (3) represents the transmitted symbol we want to estimate, and the second term represents the ISI due to the filter.

On the uplink channel, $s_u(t)$ is corrupted by additive white Gaussian noise. Thus, using the narrow band model for the noise, the input to the TWT can be expressed as

$$s'_u(t) = s_u(t) + n_{uc}(t)\cos\omega_c t - n_{us}(t)\sin\omega_c t. \tag{4}$$

$n_{uc}(t)$ and $n_{us}(t)$ represent the in-phase and quadrature components of the uplink noise, each with zero mean and variance σ_u^2. From (3) and (4)

$$s'_u(t) = r_n(t)\cos(\omega_c t + \lambda(t)). \tag{5}$$

where

$$r_n(t) = [(r(t) + n_{uc}(t))^2 + n_{us}^2(t)]^{1/2}. \tag{6}$$

$$r(t) = U(n)h(t-nT) + \sum_{i \ne n} U(i)h(t-iT), \tag{7}$$

and

$$\lambda(t) = \tan^{-1}\frac{n_{us}(t)}{r(t)+n_{uc}(t)}. \tag{8}$$

The TWT is a nonlinear memoryless amplifier. It exhibits nonlinear distortion in both the amplitude and the phase. Using a quadrature model, the output $s_d(t)$ of the TWT can be expressed in the form

$$s_d(t) = P[r_n(t)]\cos(\omega_c t + \lambda(t)) - Q[r_n(t)]\sin(\omega_c t + \lambda(t)). \tag{9}$$

From Saleh [8] an expression of P(r) and Q(r) is given by

$$P(r) = \alpha_p \frac{r}{1+\beta_p r^2} \tag{10a}$$

and

$$Q(r) = \alpha_q \frac{r^3}{(1+\beta_q r^2)^2}. \tag{10b}$$

The coefficients of (10) are obtained by a least-square error curve fitting procedure of the specific TWT characteristics, which are originally specified by the manufacturer. In Figure 2 the P(r) and Q(r) functions of (10) are plotted for $\alpha_p = 2.0922$, $\beta_p = 1.2466$, $\alpha_q = 5.529$ and $\beta_q = 2.7088$ [8]. All input and output voltages were normalized.

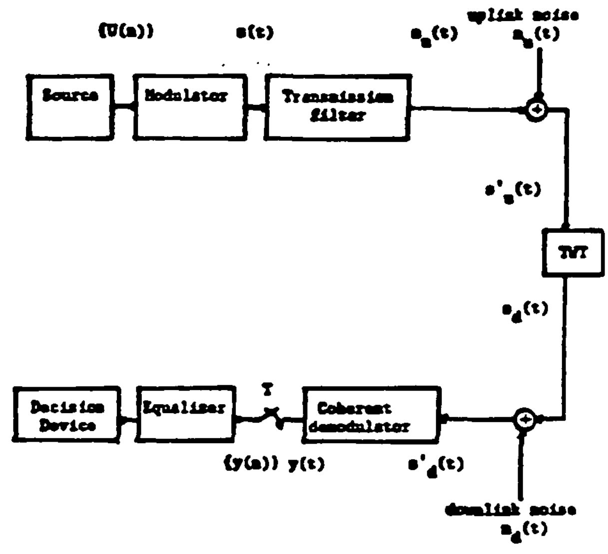

Figure 1: Satellite Communication System Model

Because of the downlink additive white noise $n_d(t)$, the received waveform $s'_d(t)$ can be expressed as

$$s'_d(t) = s_d(t) + n_{dc}(t)\cos\omega_c t - n_{ds}(t)\sin\omega_c t, \quad nT \le t \le (n+1)T. \tag{11}$$

The signal $s'_d(t)$ of (11) is now coherently demodulated by a carrier $2\cos\omega_c t$. We assume that the bandwidth of the receiving filters is wide enough so that no additional ISI distorts the signal. The output y(t) of the demodulator is sampled every T seconds to produce at the n th signaling interval the in-phase sample

$$y(n) = y(t_0) = P[r_n(t_0)]\cos\lambda(t_0) + Q[r_n(t_0)]\sin\lambda(t_0) + n_{dc}(t_0). \tag{12}$$

t_0 is an appropriately chosen sampling instant within the interval, $nT \le t \le (n+1)T$.

Under the assumption of high available power at the earth stations, the effects of the uplink noise can be considered negligible. Thus we can assume that $\lambda(t) = 0$. Then y(n) of (12) becomes

$$y(n) = P[r(t_0)] + n_{dc}(t_0). \tag{13}$$

From (7) and (13) an equivalent discrete-time model for the communication channel of Figure 1 can be represented as in Figure 3. Now, with U(n)={1,-1}, the basic relationships are

$$r(n) = A\sum_{k=-N1}^{N2} h_k U(n-k), \tag{14}$$

$$P(n) = P[r(n)] = \frac{\alpha r(n)}{1+\beta r^2(n)}, \tag{15}$$

$$y(n) = P(n) + w(n), \tag{16}$$

where α and β are specified constants that depend on the specific type of the TWT. w(n) is white Gaussian noise of zero mean and variance σ_d^2, and uncorrelated with the input data.

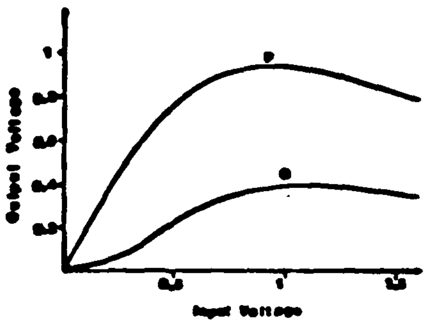

Figure 2 : TWT inphase and quadrature characteristics

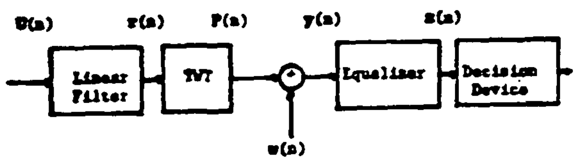

Figure 3 : Discrete Model of Satellite Communication Channel

The values of N1 and N2 represent the memory of the transmitting filter. The gain A depends on the specific operating point of the TWT.

III. THE MEAN-SQUARE ERROR EQUALIZER

Let the receiver output z(n) be expressed as the output of a Tapped- Delay Line (TDL) filter in the form of

$$z(n) = \sum_{k=-M1}^{M2} c_k y(n-k) , \tag{17}$$

where from (16), y(n)= P(n) + w(n).

In the theory of the Mean-Squares criterion, the tap weight coefficients $\{c_i\}$ of the equalizer are adjusted to minimize the mean square error

$$\epsilon^2 = E[U(n) - \sum_{k=-M1}^{M2} c_k y(n-k)]^2 . \tag{18}$$

Minimization of (18) with respect to the $\{c_i\}$ coefficients, yields the linear system of M= M1 + M2 + 1 equations

$$\sum_{k=-M1}^{M2} c_k R_y(j-k) = R_{uy}(j) , \quad j=-M1,\ldots,M2 , \tag{19}$$

where $R_y(k) = R_y(-k) = E[y(n)y(n-k)]$ and $R_{uy}(k) = E[U(n)y(n-k)]$ for all values of k.

From the uncorrelatedness of the input data and the noise, $R_{uy}(k) = R_{up}(k)$, for all values of k. Also, since the output P(n) of the nonlinearity, and the noise w(n) are independent

$$R_y(k) = R_p(k) + \sigma_w^2 \delta_{0k} . \tag{20}$$

where σ_w^2 is the variance of the noise. Thus in order to solve (19) it is necessary to evaluate first the necessary $R_p(\cdot)$ and $R_{up}(\cdot)$ coefficients. While in the case of a linear channel the calculation for the $R_p(\cdot)$ coefficients is straightforward, in the nonlinear case the evaluation may present some numerical difficulties.

Computation of the Autocorrelation coefficients

The sequence {P(n)} at the output of the nonlinearity can be considered as the output of a finite state sequential machine. Since the nonlinearity has no memory, from (14) the state sequence can be given by

$$s(n) = [U(n+N1),\ldots,U(n),U(n-1),\ldots,U(n-N2)] . \tag{21}$$

{U(n)} is an i.i.d sequence, thus {s(n)} is itself a stationary Markov chain [9].

Let us denote by Π the transition probability matrix of the Markov chain {s(n)}. A brute force evaluation of $R_p(\cdot)$ involves multiplication of square matrices of dimension $2^{N1+N2+1}$ [9-10], which would be computational impractical unless special consideration is given to the special properties of Π. In [9] a particularly effective and simple algorithm for the evaluation of autocorrelation coefficients of a nonlinear system was presented. The algorithm, as applied to our specific problem is given below.

Algorithm for the computation of $R_p(k)$

1. Let N=N1+N2+1, and store in vector β_0 (of dimension 2^N) the values at the output of the nonlinearity for each state s(j) of (21), for $j=1,2,\ldots,2^N$.

2. Compute the vector α_0 (of dimension 2^N), where the j th component is given by

$$\alpha_0(j) = \beta_0(j)/2^N , \; j=1,2,\ldots,2^N .$$

3. For k=0,1,...,N-1, do the following computations

a) $$R_p(k) = \sum_{j=1}^{2^{N-k}} \alpha_k(j)\beta_k(j) .$$

b) Store in the first 2^{N-k-1} positions of α_k, the vector α_{k+1}, computed by

$$\alpha_{k+1}(j) = \alpha_k(j) + \alpha_k(j+2^{N-k-1}) , \; j=1,2,\ldots,2^{N-k-1} .$$

c) Store in the first 2^{N-k-1} positions of of the β_k vector, the vector β_{k+1}, where

$$\beta_{k+1}(j) = \frac{\beta_k(2j-1)+\beta_k(2j)}{2} , \; j=1,2,\ldots,2^{N-k-1} .$$

4. $R_p(k) = 0$, for $k \geq N$.

The above algorithm is easy to implement and requires only two vectors of size 2^N as basic computation storage.

Computation of Cross-correlations

Since for each state s(j) of (21) the value of $P[s(j)] = \beta_0(j)$ has already been computed for the evaluation of the $R_p(\cdot)$ coefficients, a brute force technique can be easily applied for the evaluation of the cross-correlation terms. Thus from [10],

$$R_{up}(-k) = (1/2^{N-1}) \sum_{j,U(n-k)=1} P[s(j)], \quad -N1 \leq k \leq N2 , \tag{22}$$

where the summation in (22) is done over all those states where U(n-k) = 1.

In summary, the design procedure for the optimum linear MSE equalizer is given as follows. First, compute the 2^N possible values of P(n) at the output of the nonlinearity. Then use the algorithm to compute the $R_p(\cdot)$ coefficients and (22) to compute the $R_{\eta}(\cdot)$ coefficients. Finally, solve the linear system in (19). The solution of (19) yields the tap-weight coefficients of the MSE receiver.

IV. EVALUATION OF BIT-PROBABILITY OF ERROR

Unfortunately, there is no simple relationship between the residual mean square error of the MSE receiver and the bit error probability [11]. For moderate channel and equalizer memories, a brute force method that yields an exact result could be applied. Denote by D_i one of the 2^{M+N-2} possible realizations at the input of the receiver, with U(n)=1, and by c the M x 1 vector of the filter coefficients. Then from (16) and (17), the receiver output due to a specific input {U(n)} sequence is given by

$$z_i = D_i c + Wc \ , \tag{23}$$

where W is a M=M1+M2+1 row vector of noise samples.

Let {w(n)} be a white noise sequence of zero mean and variance σ_w^2. Let $\eta = Wc$. Then $E[\eta]=0$ and $\sigma_\eta^2 = \sigma_w^2 \sum_{k=-M1}^{M2} c_k^2$. Then with U(n)=1 and for a threshold of zero, the conditional error probability

$$P_e(i) = Pr[D_i c + \eta < 0/\{U(i)\}] \ , \tag{24}$$

is fixed, and

$$P_e(i) = Q(D_i c/\sigma_\eta) \ , \tag{25}$$

where

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{t^2}{2}} dt \ . \tag{26}$$

Then the average error probability P_e is given by

$$P_e = (1/L) \sum_{i=1}^{L} P_e(i) \ , \ L = 2^{M+N-2} \ . \tag{27}$$

In our numerical example the SNR is defined as

$$SNR = 10 \log_{10}(P_{av}^2 / 2\sigma_\eta^2) \ , \tag{28}$$

where $P_{av} = (1/L) \sum_{i=1}^{L} P[r^{(i)}(0)]$.

If the exact error probability of (27) proves too cumbersome and too time consuming to evaluate because of the large number of terms, one can resort to a number of different approximate methods that yield tight upper and lower bounds of P_e [11].

V. NUMERICAL EXAMPLE

The purpose of this section is to illustrate the application of our results in the design of a linear optimal receiver, and to compare its performance with other receivers for a digital satellite link.

In our model of the linear part of the satellite link, we assume that the ISI is introduced by a 3-pole Butterworth filter. The two sided bandwidth B of the filter is the same as the minimum Nyquist rate (i.e., BT=1). The number of samples considered for the ISI is determined by those ISI samples whose magnitude are at least greater than 0.01 times the main sample. In our example, a channel memory (N1+N2) of 3 ISI terms was considered adequate.

The characteristics of the TWT for this study are the same as those in Figure 2. Thus in the evaluation of P[r(n)] in (15), the parameters of the TWT are $\alpha = 2.0922$ and $\beta = 1.2466$. As mentioned before, those values were taken from Saleh ([8], Figure 5) and represent a specific satellite TWT. The gain factor A_o of (14) was determined so that with no ISI the TWT would operate at the 2 dB input backoff point. Because of the low ISI introduced by the transmission filter, a 4-tap (M1+M2+1=4) TDL linear receiver was considered to be adequate. Thus, $L = 2^{(4+4-2)} = 64$.

Now we compare our optimum linearly equalized MSE receiver with the conventional linear receivers. Using the brute force technique described in Section 4, the bit error probability for the various receivers was evaluated and plotted in Figures 4 and 5, for values of SNR as defined in (28).

Figure 4 exhibits the P_e performance of the designed MSE filter, and the P_e performance of two 3-pole Butterworth receiving filters. One receiving filter (with BT=1) is identical to the transmitting filter, while the other one has BT=0.75. In Figure 5, the performance of the M.S receiver is compared with that of two 4-pole Butterworth receiving filters. One has BT=0.75 and the other one has BT=1. A numerical search procedure for Butterworth filters with different BT products, showed that an increase in BT does not necessarily correspond to an improved P_e performance. In fact, filters with BT=2 are only marginally better than filters with BT=1.

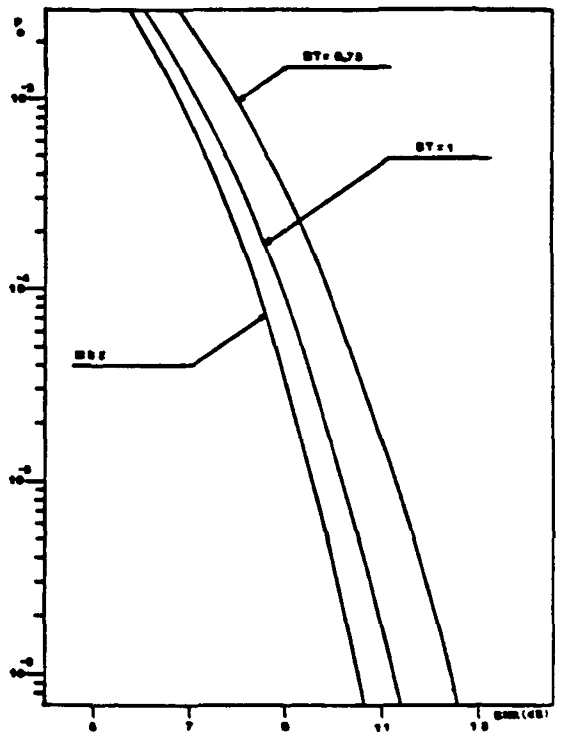

Figure 4 : P_e performance of MSE and 3-pole Butterworth receiving filters.

For P_e = 10E-6, the optimum MSE receiver is at least 0.8 dB better than the 3-pole Butterworth filters and 1.2 dB better than the 4-pole Butterworth filters. The P_e performance of a channel with no ISI, but with the identical TWT, carrier power and noise variance was evaluated. The numerical results showed that for these examples the bit error rate of the MSE equalizer is very close to that of the no ISI case [10].

VI. SUMMARY AND CONCLUSIONS

In this paper we considered the problem of modeling and equalization of a digital satellite nonlinear and bandlimited channel. Starting from a typical satellite link, we developed the corresponding BPSK discrete-time model, and solved for the optimum linear MSE receiver. A simple and computationally efficient algorithm was derived for the evaluation of the equalizer coefficients, based on the memoryless nonlinearity of the system. Numerical examples for a typical satellite link demonstrated that the optimum linear MSE receiver outperforms the conventional linear type receiving filters. In general, our modeling and equalization techniques provide a simple and computationally efficient alternative to existing approaches.

REFERENCES

1. S.A Fredricsson, *Optimum Receiver Filters in Digital Quadrature Phase-Shift- Keyed Systems with a Nonlinear Repeater*, IEEE Trans. on Comm., Vol. COM-23, No.12, pp.1389-1399, Dec. 1975.

2. M.F.Mesiya, P.J. McLane and L.L. Campbell, *Maximum Likelihood Sequence Estimation of Binary Sequences Transmitted Over Bandlimited Nonlinear Channels*, IEEE Trans. on Comm., Vol. COM-25, No. 7,pp. 633-643, July 1977.

3. M.F.Mesiya, P.J. McLane and L.L Campbell, *Optimal Receiver Filters for BPSK Transmission over a Bandlimited Nonlinear Channel*, IEEE Trans. on Comm., Vol. COM-26, pp.12-22, Jan. 1978.

4. E.Biglieri, M.Elia and L. L.Presti, *Optimal Linear Receiver Filter for Digital Transmission over Nonlinear Channels*. Proc. 1983 Intern. Tirrenia Workshop on Dig. Commun., pp. F.3.1-F.3.13, Sept. 1983.

5. E.Ekanayake and D.P. Taylor, *A Decision Feedback Receiver Structure for Bandlimited Nonlinear Channels*, IEEE Trans. on Comm., Vol. COM-29, No. 5, pp.539-548, May 1981.

6. S.Benedetto and E.Biglieri, *Nonlinear Equalization of Digital Satellite Channels*, IEEE Journal on Select. Areas in Comm., Vol. SAC-1, No.1, pp.57-62, Jan. 1983.

7. S.Benedetto, E. Biglieri, R. Daffara, *Modeling and Performance Evaluation of Nonlinear Satellite Links- A Volterra Series Approach*. IEEE Trans. on Aerospace and Elec. Systems, Vol. AES-15, No.4, pp.494-506, July 1979.

8. A.A.M. Saleh, *Frequency -Independent and Frequency-Dependent Nonlinear Models of TWT Amplifiers*, IEEE Trans on Comm., Vol. COM-29, No 11, pp 1715-1720., Nov. 1981.

9. R.Padovani and G.L.Pierobon, *Spectral Analysis of Digital Messages Through Finite-Memory Transformations*, IEEE Trans. on Comm., Vol. COM-32, No. 11, pp. 1214-1218, Nov. 1984.

10. K.Konstantinides, *Channel Modeling and Equalization Algorithms Based on Least Squares Techniques*, Ph.D Dissertation, Un. of Calif., Los Angeles, 1985.

11. J.G. Proakis, *Digital Communications*, Mc. Graw-Hill, 1983.

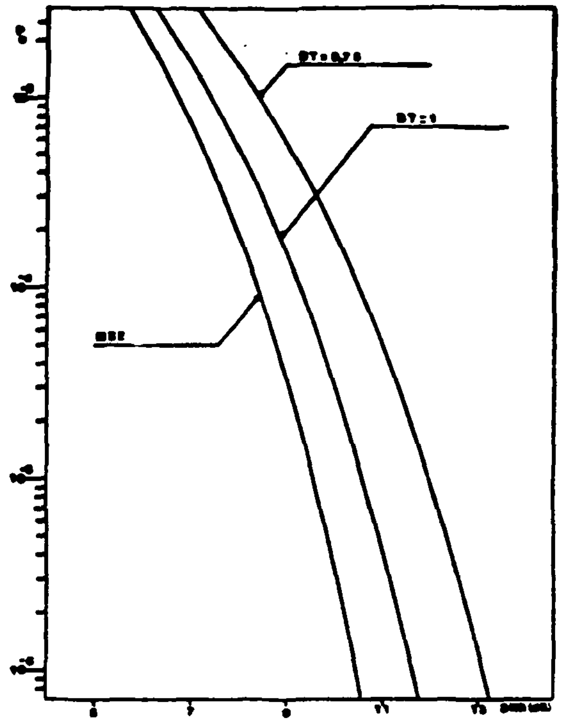

Figure 5 : P_e performance of M.S.E and 4-pole Butterworth receiving filters.

N86-30072 D3 10P.

18335

CD 146017

On Realizations of Least-Squares Estimation and Kalman Filtering by Systolic Arrays

M.J. Chen and K. Yao

1. INTRODUCTION

Least-squares (LS) estimation is a basic operation in many signal processing problems. Given y=Ax+v, where A is a mxn coefficient matrix, y is a mx1 observation vector, and v is a mx1 zero mean white noise vector, a simple least-squares solution is finding $\hat{x}$ which minimizes $\|Ax-y\|$. It is well known that for an ill-conditioned matrix A, solving least-squares problems by orthogonal triangular (QR) decomposition and back substitution has robust numerical properties under finite word length effect since 2-norm is preserved. Many fast algorithms have been proposed and applied to systolic arrays. Gentleman-Kung (1981) first presented the triangular systolic array for a basic Givens reduction. McWhirter (1983) used this array structure to find the least-squares estimation errors. Then by geometric approach, several different systolic array realizations of the recursive least-squares estimation algorithms of Lee et al (1981) were derived by Kalson-Yao (1985). We consider basic QR decomposition algorithms and find that under one-row time updating situation, the Householder transformation degenerates to a simple Givens reduction. Next, we derive an improved least-squares estimation algorithm by considering a modified version of fast Givens reduction. From this approach, the basic relationship between Givens reduction and Modified-Gram-Schmidt transformation can easily be understood. We also can see this improved algorithm has simpler computational and inter-cell connection complexities while compared with other known least-squares algorithms and is more realistic for systolic array implementation.

Minimum variance estimation (popularized by Kalman (1960)) is the generalized form of a least-squares problem, where the state vector x is characterized by the state equation $x_{k+1}=Fx_k+w$, the system noise w and the observation noise v are colored. The original algorithm presented by Kalman can have poor numerical property. Some algorithms for improving numerical properties, such as square-root covariance and square-root information methods have been studied. Now, we find that after the whitening processing, this minimum variance estimation can be formulated as the modified square-root information filter and be solved by the simple least-squares processing. This new approach contains advantages in both numerical accuracy as well as computational efficiency as compared to the original Kalman algorithm. Since all these processings can be implemented by systolic arrays, high throughput rate computation for Kalman filtering problems become feasible.

2. SIMPLE LEAST-SQUARES ESTIMATION

Given the equation $b=Ax+v$, it is well known that we can solve the least-squares solution $\hat{x}$ by normal equation. However, this approach not only requires the computation of a matrix inverse but also doubles the condi tion number when we form $A'A$. Although using singular value decomposition for least-squares solution can improve numerical properties, the computa tional complexity involved in SVD is not low. Besides, fast algorithm for SVD is still underdevelopment. Lattice structure for least-squares solu tion was proposed and studied by Lee et al (1981). This approach was shown to have stable numerical property and regular hardware structure. However, this method required shifting property of the coefficient matrix and can not apply to all general cases. QR decomposition is another solu tion to obtain $\hat{x}$, since 2-norm is preserved by multiplying an orthogonal matrix Q, then by letting $QA=R$ be a upper triangular matrix, the $\hat{x}$ can be obtained by using back substitution for the equation $Rx=b$. This approach has robust numerical properties since the 2-norm is fixed, the rounding error caused by finite word length effect will not grow. Basically, there are three ways for performing QR decomposition, namely, Householder trans formation, Givens reduction, and Modified-Gram-Schmidt orthogonalization. It can be shown that under one row time updating situation (as in the sys tolic array implementation), the Householder transformation matrix will degenerate to a simple Givens reduction case.

Systolic array implementation for QR decompositions in least-squares esti mation was first explored by Gentleman-Kung and followed by McWhirter and Kalson-Yao. By using a triangular systolic array, it was shown that the estimation error for the last observation can be solved at every clock period. The systolic array structure for least-squares estimation is shown in Figure 2.1. To achieve fully pipelined operation, the input rows are skewed and propagated like wavefronts in the diagonal direction. There are only two basic processing units, boundary cell and internal cell, are required by this systolic array. Communication between different process ing units are all local. The properties of regularity and local communi cation are consistent with the philosophy of VLSI implementation. Summary of input/output formats and operation functions for two kinds of process ing units are shown in Table 1 and Table 2 respectively.

Table 1. Input/Output format of systolic array algorithms

	BI_1	BI_2	BO_1	BO_2	II_1	II_2	IO_1	IO_2
Givens	σ	x	σ'	d/d',x/d'	d/d',x/d'	b	d/d',x/d'	b'
F-Givens	σ	x	σ'	d/d',σx/d' x	d/d',σx/d' x	b	d/d',σx/d' x	b'
M-G-S(I)	σ	x,e	σ'	x/d,x/(1-σ) d	x/d,x/(1-σ) d	b,e	x/d,x/(1-σ) d	b',e'
M-G-S(II)	σ	x	σ'	x/d',x/(1-σ)	x/d',x/(1-σ)	b	x/d',x/(1-σ)	b'

The above symbols are for notations only, their physical meaning may change for different algorithms.

Table 2. Operational functions of processing units

	Boundary cells	Internal cells
Givens	$d'=(d^2+x^2)^{1/2}$ $BO_2 \leftarrow d/d',\ x/d'$ $\sigma' \leqq (d/d')*\sigma$	$b'=(d/d')*b - (x/d')*k$ $k'=(x/d')*b + (d/d')*k$
F-Givens	$d'=d + (\sigma*x)*x$ $BO_2 \leftarrow (\sigma*x)/d',\ d/d'$ $\sigma' \leqq \sigma*(d/d')$	$b'=b - x*k$ $k'=(d/d')*k + (\sigma*x/d')*b$
M-G-S(I)	$d=e$ $BO_2 \leftarrow x/d,\ x/(1-\sigma)$ $\sigma' \leqq \sigma + (x/d)*x$	$k'=k + b*(x/(1-\sigma))$ $b'=b - k'*(x/d)$ $e'=e - k'^2/d$
M-G-S(II)	$d'=d + (x/(1-\sigma))*x$ $BO_2 \leftarrow x/d',\ x/(1-\sigma)$ $\sigma' \leqq \sigma + (x/d')*x$	$k'=k + b*(x/(1-\sigma))$ $b'=b - k'*(x/d')$

From systolic array point of view, the difference between algorithms proposed by McWhirter and Kalson-Yao lies in the basic computations in two kinds of processing units. Since these algorithms were derived from two different approaches, specifically Givens reduction and Modified-Gram-Schmidt orthogonalization, the basic relationship for these two QR decomposition methods under one row time updating can be compared as follows. First, we derived the modified expression for the fast-Givens reduction as given by

$$Q \begin{bmatrix} (1/\sqrt{d})d, & (1/\sqrt{d})dk_2, & \ldots & (1/\sqrt{d})dk_k \\ \sqrt{\sigma}x, & \sqrt{\sigma}b_2, & \ldots & \sqrt{\sigma}b_k \end{bmatrix}$$

$$= \begin{bmatrix} (1/\sqrt{d'})d', & (1/\sqrt{d'})d'k_2', & \ldots & (1/\sqrt{d'})d'k_k' \\ 0, & \sqrt{\sigma}'b_2', & \ldots & \sqrt{\sigma}'b_k' \end{bmatrix},$$

the updating equation for this modified-fast-Givens algorithm becomes,

$$\begin{array}{lll} \text{Boundary cell:} & d'=d + x^2/(1/\sigma) & (1/\sigma')=(1/\sigma) + x^2/d \\ \text{Internal cell:} & b'=b - (x/d)*dk & d'k'=dk + b*x/(1/\sigma) \end{array} \qquad [1]$$

By comparing the computational complexity between the fast Givens algorithm by Gentleman (1973) and that in [1], we can see [1] has one multi plication less than the original algorithm. And since we do not have interest on the real rotated elements like $(1/\sqrt{d})dk_k$, we do not have the risk of dividing by a very small d. The numerical properties of the modified algorithm is then expected to comparable to the numerical properties of the original one. By equation [1], the basic duality associations between Givens reduction and Modified-Gram-Schmidt orthogonalization is summarized in Table 3, which allows us to derive different algorithms for least-squares estimation from different approaches with efficiency.

Table 3. Duality association for M-G-S and Fast-Givens reduction.

M-G-S(II)	k_{mgs}	σ_{mgs}	x_{mgs}	b_{mgs}	d_{mgs}
F-Givens	$d_{fg}*k_{fg}$	$1-\sigma_{fg}$	$\sigma_{fg}*x_{fg}$	$\sigma_{fg}*b_{fg}$	d_{fg}

With systolic array implementation, comparison of computational complexity for algorithms discussed above can be made by comparing the number of operations required in each processing unit. When the dimension of the coefficient matrix becomes large, wavefront array processing of Kung (1983) becomes more appropriate for the control scheme. In this case, the speed of this "wavefront" will be decided by the slowest processing unit along each wavefront. In modified fast Givens algorithm, equations for boundary cell are non-recursive and can be done in parallel if we can double the computational capability of each boundary cell. In this case, the wavefront speed and then the throughput rate can be doubled. The systolic array we discussed above will generate estimation error at each clock period. While the estimated vector $\hat{x}$ is not shown explicitly, $\hat{x}$ can be solved by back substitution which can be done by just appending a nxn identity matrix after the coefficient matrix A.

3. MINIMUM VARIANCE ESTIMATIONS AND KALMAN FILTERING

Often the signal vector x is a random process and can be modeled as a first order recursive equation. In this case, a first order recursive estimation (or Kalman filtering) problem can be stated as follows,

$$x_{k+1}=Fx_k+w_k, \quad y_k=Cx_k+v_k, \qquad [2]$$

where F and C are time-varying coefficient matrices with dimension nxn and mxn respectively. w_k is a nx1 and v_k is a mx1 zero mean noise vectors with known covariance matrices W_k and V_k respectively. It is assumed that noises w and v are uncorrelated and $E[w_i w_j]=E[v_i v_j]=0$ for all $i \neq j$. Under the minimum variance criterion, we want to find $\hat{x}_k$ for all k, such that $E\|(x_k-\hat{x}_k)^2\|$ is minimized. Kalman showed that $\hat{x}_k$ can be obtained by the recursive algorithm given as

$$
\begin{aligned}
&\hat{x}_k = F\hat{x}_{k-1} + K_k[y_k - CF\hat{x}_{k-1}],\\
&K_k = \tilde{P}_k C'[C\tilde{P}_k C' + V]^{-1},\\
&\text{where } \tilde{P}_k = F\tilde{P}_{k-1}F' + W,\\
&P_k = \tilde{P}_k - K_k C\tilde{P}_k.
\end{aligned}
\qquad [3]
$$

The information matrix is defined as the inverse of the error covariance matrix P. Besides [3], it is shown that instead of propagating the error covariance matrix, the Kalman filtering problem can be solved by propagating the information matrix during the iterations. Both covariance and information filters are recursive since the current updating depends only on results from previous stage. The choice between covariance filter and information filter depends on the values of n and m. When $n>m$, which is usually the case, the original Kalman filtering is chosen to avoid the inverse of the $n \times n$ matrix. However, Kalman algorithm is known for its poor numerical properties, especially for non-observable coefficient matrices. The original Kalman filter needs an approximate $O(n^3)$ multiplication time for each iteration. If $m>1$, computation of a matrix inversion is inevitable. Since all equations are sequential in manner, if real time computation is required for a Kalman filtering problem, some modifications must be done to insure the capability for parallel computation. Among many possible modified algorithms, square-root filtering have been proved to have computational efficiency and robust numerical properties under finite word length effect (Kaminski 1971). The main advantage of the square root filter is that we can handle the covariance matrix by its square root form which has condition number smaller than the original one. Therefore, for ill-conditioned problems, when we used the square root filter with a single precision machine, we can expect the same numerical result as if we have used the original algorithm on a double precision machine. Updating processings for both square root covariance filter and square root information filter can be expressed in matrix forms and handled by the QR decomposition method which is capable of systolic array implementation. However, only square-root information filter allows us to update the estimated state vector as well as the information matrix by using the same transformation matrix Q. When both updated covariance matrix and state vector are important to us, we find square-root information filter is a better solution for the systolic array implementation. The square-root information filter requires computation of the inverse of the coefficient matrix F, which will cause bad numerical properties for F being near singular. One version of the square root information matrix method for Kalman filtering was considered by Paige and Saunders (1977). It is shown that by using whitening processing through Cholesky decomposition, the Kalman filtering can be represented as a simple least-squares problem. This approach does not require the computation of the inverse of the matrix F and is more suitable for systolic array implementation.

The whitening processing can be briefly described as below. Assume $W=\tilde{L}_w\tilde{L}_w'$ and $V=\tilde{L}_v\tilde{L}_v'$ are the Cholesky decomposition of covariance matrices W and V. With $W^{-1}=L_wL_w'$ and $V^{-1}=L_vL_v'$, it can be proved that $L_w=\tilde{L}_w^{-T}$ and $L_v=\tilde{L}_v^{-T}$. $\tilde{w}_k=L_w'w_k$ and $\tilde{v}_k=L_v'v_k$ are whitened noises with identity covariance matrices.

Denote $\tilde{F}=L_w'F$, $\tilde{C}=L_v'C$, and $\tilde{y}_k=L_v'y_k$. We can express the whitened system equations in the matrix-vector form as

$$
\begin{bmatrix} 0 \\ \tilde{y}_1 \\ 0 \\ \vdots \\ \vdots \\ \tilde{y}_k \end{bmatrix} = \begin{bmatrix} -L_w' & & & 0 \\ \tilde{C}^w & & & \\ \tilde{F} & -L_w' & & \\ & \tilde{C}^w & & \\ & & \ddots & \\ 0 & & & \tilde{C} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_k \end{bmatrix} + \begin{bmatrix} \tilde{w}_1 \\ \tilde{v}_1 \\ \tilde{w}_2 \\ \vdots \\ \tilde{v}_k \end{bmatrix} \qquad [4]
$$

Since the noise vector in [4] has zero mean and identity covariance matrix, we can get $X_{min}=[\hat{x}_1,\ldots\hat{x}_k]$ by solving [4] as a LS problem. After applying QR decomposition to [4] at time k, we have

$$
\begin{bmatrix} R_1 & R_{12} & & & 0 \\ & R_2 & R_{23} & & \\ \cdot & & & & \\ \cdot & & & & \\ 0 & & & R_{k-1} & R_{k-1,k} \\ & & & & \bar{R}_k \end{bmatrix} \begin{bmatrix} x_{1,k} \\ x_{2,k} \\ \cdot \\ \cdot \\ x_{k-1,k} \\ x_{k,k} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ y_{k-1} \\ \bar{y}_k \end{bmatrix} \qquad [5]
$$

We can see that R_i, i=1,2...k, in [5] are all upper triangular matrices, and $\hat{x}_k$, the optimum estimated vector at time k, depends only on the last line, i.e., $\bar{R}_k\hat{x}_k=\bar{y}_k$. Furthermore, at T=k+1, the updating equation depends on the last row of [5] only. That is, the QR decomposition at T=k+1 only depends on a (2n+m)x(2n+1) matrix as in [6]. When the QR decomposition of [6] is completed, we have $\bar{R}_{k+1}$ (upper triangular) and $\bar{y}_{k+1}$ ready for iteration of next stage.

$$
Q \begin{bmatrix} \bar{R}_k & 0 & \bar{y}_k \\ \tilde{F} & -L_w' & 0 \\ 0 & \tilde{C}^w & \tilde{y}_k \end{bmatrix} = \begin{bmatrix} R_k & R_{k,k+1} & y_k \\ 0 & \bar{R}_{k+1} & \bar{y}_{k+1} \\ 0 & 0 & * \end{bmatrix} \qquad [6]
$$

where * is the term used to compute the residue.

The upper triangular matrix $\bar{R}_k$ can be shown to be the square-root of the inverse of the error covariance matrix $P_k=E[(x_k-\hat{x}_k)(x_k-\hat{x}_k)']$. That is, this algorithm, which propagates the square root information matrix for next iteration, is actually a modified square-root information filtering.

4. SYSTOLIC ARRAY IMPLEMENTATIONS FOR KALMAN FILTERING

From last section, we can see that the basic operations for square root Kalman filtering can be described in two parts. The first one, whitening processing includes operations such as Cholesky decomposition, inverse of triangular matrix, and matrix multiplication. Secondly, the QR decomposition is applied. Obviously, these two parts can be operated in parallel. That is, we can start the whitening processing for the (k+1)st iteration as well as the QR decomposition for the k-th iteration at the same time in a pipelined manner.

The original square-root information filter involves the computation of the inverse of the coefficient matrix F which not only increases the computational complexity but also causes bad numerical properties when coefficient matrix F is singular or near singular. This shortcoming can be recovered by choosing the modified square root information filtering in [4]. As shown from [4]-[6], formulation of the modified square-root information filter involves only multiplication between coefficient matrices and the inverse of the square root noise covariance matrices. For noise with positive definite covariance, square root covariance matrix always exists.

4.1 Whitening Processing

The whitening processing is done by multiplying the coefficient matrix with a whitening operator L' where $(LL')^{-1}$ is the given covariance matrix of the additive noise. Since a covariance matrix is a positive definite symmetric matrix, the square root matrix can be obtained by the Cholesky decomposition. A triangular systolic array for Cholesky decomposition is designed for this purpose with outputs skewed to match the input format of the QR systolic array.

The inversion of a upper triangular matrix is simple after we built the basic systolic array for QR decomposition. The idea for the inversion of a upper triangular matrix is the same as solving the back substitution.

With $UU^{-1}=I$, let $U^{-1}=[\underline{u}_1, \underline{u}_2, \ldots \underline{u}_n]$, with $\underline{u}_i$ being a nx1 column vector. A matrix inversion can be divided into n sets of linear equations, each having the form of $U\underline{u}_i=\underline{e}_i$, i=1,2,...n, where $\underline{e}_i$ is a nx1 column vector with i^{th} element equals to 1, and all others being 0, and can be solved by a systolic array.

4.2 QR Decomposition for Kalman Filtering

Equation [6] suggests that $\hat{x}_k$ can be solved as a least-squares solution by a 2nx2n QR systolic array. However, serious delay will be caused by the fact that R_k and $\bar{R}_{k+1}$ are not in-place computations. That is, we have trouble to move the newly formed $\bar{R}$ from the upper-right corner to the lower-left corner in our triangular array for the next iteration. That is, the computation at stage k+1 can not start until the last element of R_k is completed. In this "waiting" period, most of processing units are idle and the pipeline is empty. It will cause delay for at least 2n clock periods.

This disadvantage can be overcome by in-place computations for $\bar{R}_k$ and R_{k+1}. This can be done by partitioning the original matrix into two strips, and perform the partitioned QR decomposition by the systolic array structure proposed in Figure 2. In this approach, a nxn QR systolic array as well as a rotation array which consists of nx(n+1) internal cells are used. Once elements of R_k are formed, it is ready to be used for computations at stage k+1. Here we need only to pass transformed elements generated by the first strip to the rectangular rotation array for the pre-processing of the second strip. This input format is shown in Figure 3. Since all these can be done in fully pipelined manner and in-place computations are obtained, complicated inter-cell connection and control scheme can both be avoided. To obtain the estimated value $\hat{x}_k$, we can just append an identity matrix I after the second strip, and we get result every 3n+m clock periods.

5. CONCLUSION

In this paper, we first survey existing algorithms for least-squares estimations by systolic arrays. Basic comparisons are made based on computation and inter-cell connection complexities of elementary units. Finally, by choosing the square-root information filtering algorithm, we showed a simple way to solve the Kalman filtering as a least-squares problem that can be processed by systolic arrays. Systolic array for Cholesky decomposition is also proposed for whitening processing. By manipulating the data properly, the Kalman filtering can be processed under fully pipelined manner. There is no special constraint on our system equations and standard time-varying coefficient matrices and non-stationary colored noises are assumed in our model. Most of the processing units we need for this square root information filter do not involve square-root computations. The only exception is the computations for the Cholesky decomposition. However, for pipelined operation between whitening processing and QR decomposition, the later certainly involved more computational work than the former. Since there is only n square-root computations required in each iteration as compared with the operations required for QR decomposition, Cholesky decomposition will not become the bottleneck for this algorithm. For many real life problems where we can assume noises are stationary, then covariance matrices W and V are fixed during our operation. In this case, inversed square-root covariance matrices can be obtained by pre-processing and our Kalman filtering can be solved as a simple least-square problem. Since all operations can be performed by the designed systolic array processing, which have the input/output formats matched to each other, the entire hardware design can be viewed as a pipelined structure. The estimated vector can be obtained with the O(n) in time while compared with the $O(n^3)$ for the original Kalman filter. Finally, since this is a square root matrix operation, good numerical property can also be obtained.

This work is partially supported by NASA/Ames Research Contract NAG-2-304

REFERENCES

Bierman G 1977 Factorization Methods for Discrete Sequential Estimation, Academic Press, NY

Bierman G 1982 Square-Root Information Filtering and Smoothing for Precision Orbit Determination, Math. Programming Study 18

Bjorck A 1967 Solving Linear Least Squares Problems by Gram-Schmidt Orthogonalization, BIT 7 1-21

Duncan D B and Horn S D 1972 Linear Dynamic Recursive Estimation from the Viewpoint of Regression Analysis, J. ASA 67 815-821

Dyer P and McReynolds S 1969 Extension of Square-Root Filtering to Include Process Noise, J. Opt. Theory and Appl. 6 444-459

Gentleman W M 1975 Error Analysis of QR decompositions by Givens Transformations, Linear Algebra and Its Applications, 10 189-197

Gentleman W M 1973 Least Squares Computation by Givens Transformations Without Square Roots, J. Inst. Math. Appl. 12 329-336

ORIGINAL PAGE IS
OF POOR QUALITY

Gentleman W M and Kung H T 1981 Matrix Triangularization by Systolic Arrays, Proc. SPIE, Real-Time Signal Processing IV, 298 19-26

Hammarling S 1974 A Note on Modifications to the Givens Plane Rotation, J. Inst. Maths Appl. 13 215-218

Kalman R E 1960 A New Approach to Linear Filtering and Prediction Prob lems, J. Basic Engineering, 82 35-45

Kalson S and Yao K 1985 Systolic Array Processing for Order and Time Recursive Generalized Least-Squares Estimation, Proc. SPIE, Real-Time Signal Processing VIII, 564 28-38

Kaminski P G et al 1971 Discrete Square Root Filtering: A Survey of Cur rent Techniques, IEEE Tran. on Auto. Control, 6 727-736

Kung H T 1982 Why Systolic Arrays, IEEE Computer, 15 37-46

Kung S Y 1983 VLSI Design for Massively Parallel Signal Processors, Micro processors and Microsystems

Lawson C and Hanson R 1974 Solving Least Squares Problems, Prentice-Hall, NJ

Lee D T, Morf M and Friedlander B 1981 Recursive least square ladder esti mation algorithms, IEEE Tran. ASSP, 3 627-641

Ling F and Proakis J 1984 A Recursive Modified Gram-Schmidt Algorithm with Application to Least Squares Estimation and Adaptive Filtering, Int. Sym. on Circuit and System

Mead C and Conway L 1980 Introduction to VLSI systems, Addison-Wesley, Mass.

McWhirter J G 1983 Recursive least-Squares Minimization Using a Systolic Array, Proc. SPIE, Real-Time Signal Processing VI, 431 103-112

Paige C C and Saunders M A 1977 Least Squares Estimation of Discrete Lin ear Dynamic Systems Using Orthogonal Transformation, SIAM J. Numer. Anal., 14 180-193

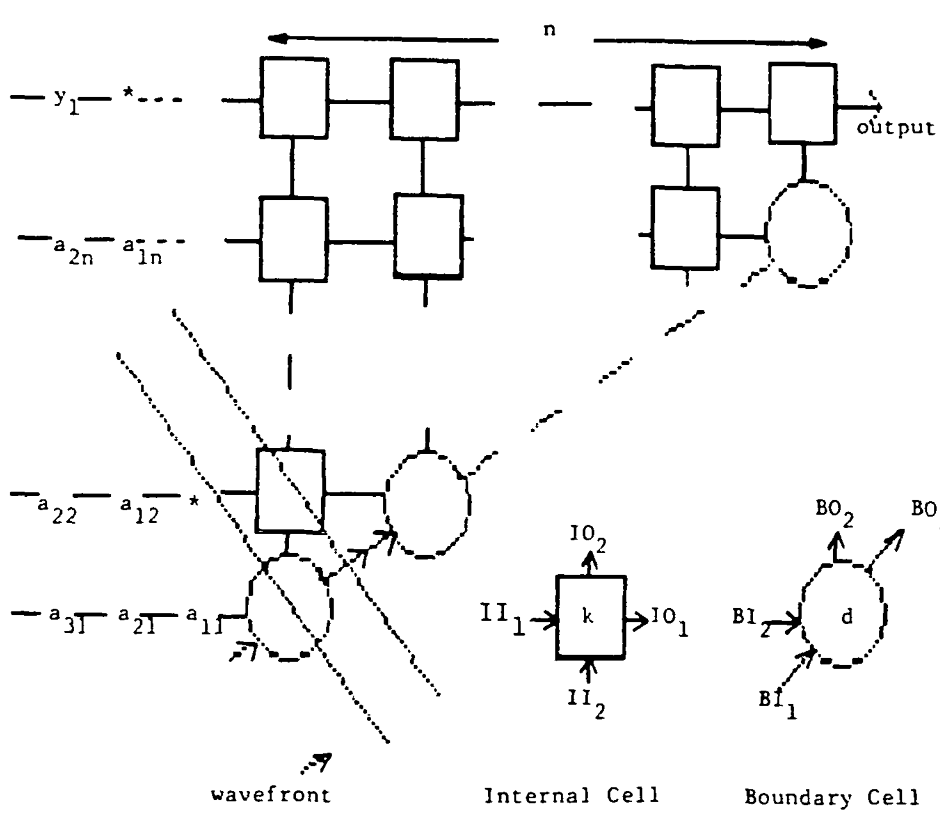

Figure 1: Systolic array for least-squares estimation.

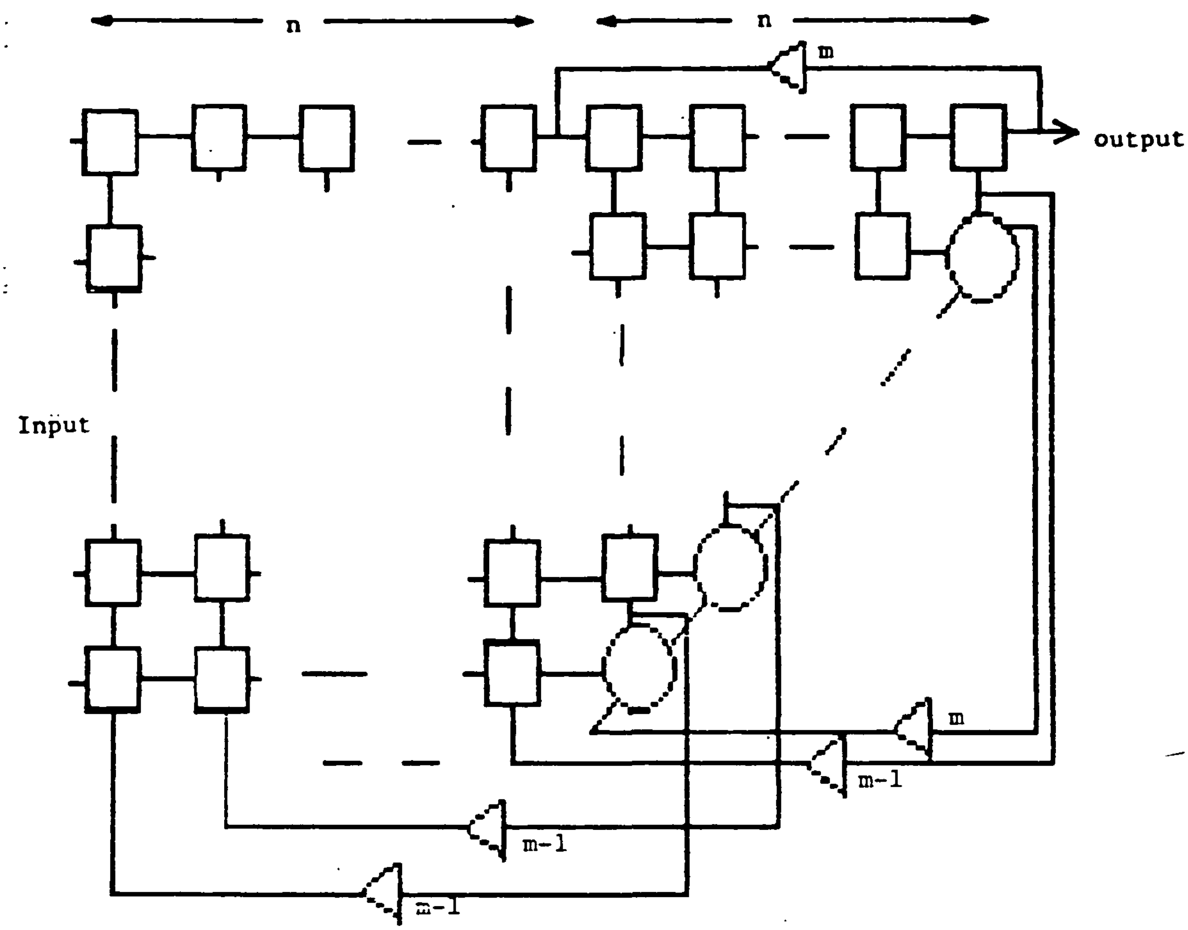

Figure 2: QR systolic array for Kalman filtering

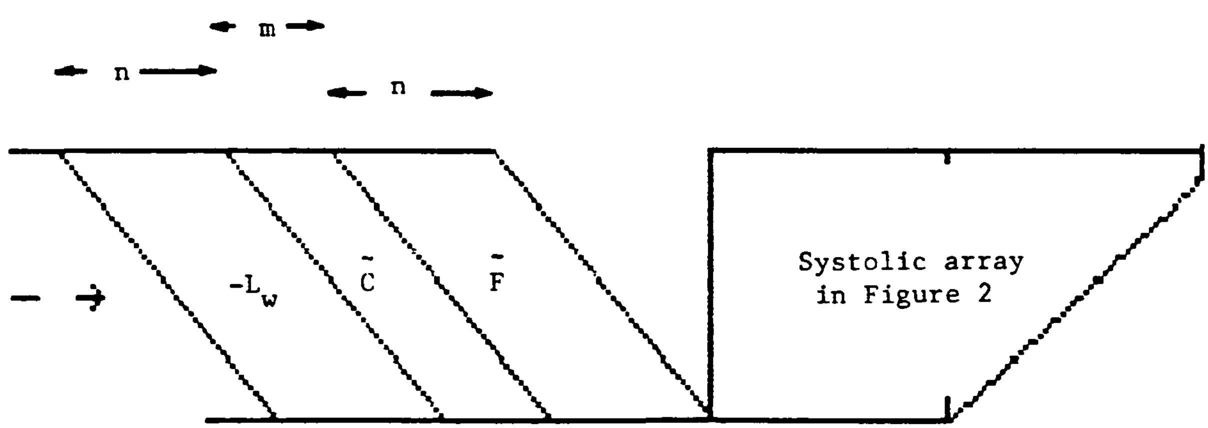

Figure 3: Input format for systolic array Kalman filtering