論文95-32B-10-9

QR 분해와 채널 분해법을 이용한 비선형 격자 알고리듬

(Nonlinear Lattice Algorithms using QRD and Channel Decomposition)

安 奉 萬 *, 白 興 基 **

(Bong-Man An and Heung-Ki Baik)

요 약

본 논문에서 쌍선형 필터를 등가의 선형 다채널 필터로 변환하여 QR 분해법을 이용한 쌍선형 순환 최소 자승 격자 알고리듬을 유도하였다. 입력 벡터의 채널 분해법을 이용하여 순방향과 역방향 입력 벡터의 차수 순환 관계를 정의하였다. 순방향과 역방향 입력 벡터를 이용하여 순방향과 역방향 데이터 행렬을 정의하였고 데이터 행렬의 직교화는 QR 분해법을 이용하였다. 이 채널 분해 방법을 이용하여 얻어진 쌍선형 격자식은 Baik에 의하여 유도된 격자식과 같게 된다. Baik에 의하여 제안된 직접형의 계수 변환 알고리듬을 이용할 수 있다. 식오차 알고리듬과 출력오차 알고리듬으로 QRD 분해법을 이용한 쌍선형 순환 최소 자승 격자 알고리듬을 유도하였으며, 제안한 알고리듬의 성능을 평가하기 위하여 쌍선형 시스템의 시스템 식별 문제에 적용하였다.

Abstract

In this paper, we transformed the bilinear filter into an equivalent linear multichannel filter and derived QR decomposition based recursive least squares algorithms for bilinear lattice filters. We also defined order update relation of the forward and the backward input vectors by using the channel decomposition. The forward and the backward data matrices were defined by using the forward and the backward input vectors and orthogonalized with the QR decomposition. we can obtain the lattice equations of the bilinear filters by using the channel decomposition. we can be derived the lattice equations of the bilinear filters using this decomposition process which are the same as the lattice equations derived by Baik, we can use the coefficient transformation algorithm proposed by Baik. We derived the equation error and the output error algorithm of the QRD based RLS bilinear lattice algorithm. Also, we evaluated the performance of the proposed algorithms through the system identification of the bilinear system.

I. 서 론

최소 평균 자승 알고리듬은 구조의 간단성과 구현하기 쉬운 장점으로 인하여 많은 응용 분야에 사용되고 있으나 하드웨어 기술의 발전으로 우수한 수렴 특성과 궤적 능력을 지닌 최소 자승 알고리듬이 최소 평균 자승 알고리듬에 비하여 각광을 받게 되었다. 그러나 최소 자승 알고리듬은 최소 평균 자승 알고리듬에 비하여 계산 량이 많다는 단점을 지니고 있다.

비선형 필터는 과거에 취급할 수 없는 분야에 대한 새로운 관심과 선형 필터로 처리할 수 없는 분야 등이 생김에 따라 비선형 필터의 연구도 활발히 진행되어 왔다[1-3]. 대표적인 비선형 필터로는 Volterra 급수

* 正會員, 井邑工業專門大學 電子通信科
(Dept. of Elec. Comm., Jung Eup Indust. College)
** 正會員, 全北大學校 電子工學科
(Dept. of Elec. Eng., Chonbuk Nat'l Univ.)
接受日字: 1995年1月3일, 수정완료일: 1995年10月5日

에 바탕을 둔 Volterra 필터와 쌍선형(bilinear) 필터가 있다. Volterra 필터는 많은 수의 커널들을 포함하고 있기 때문에 절단된 2차 형태의 Volterra 필터를 주로 사용하고 있다. 그러나 Volterra 필터와 쌍선형 필터는 선형 필터에 비하여 많은 수의 계수를 포함하고 있고 이에 따라 많은 량의 계산을 필요로 한다.

본 논문은 쌍선형 적응 격자 필터링 과정을 유도하는 논문이다. 많은 물리적인 시스템의 비선형 모델은 주로 Volterra 시스템을 이용하여 수행되어 왔다. 그러나 IIR 필터가 FIR 필터보다 더 적은 수의 계수를 가지고 많은 선형 시스템을 모델링 할 수 있는 것처럼 비선형 시스템에서도 안정도가 보장되는 경우에 Volterra 시스템에 비하여 적은 수의 계수로서 비선형 시스템을 쌍선형 시스템으로 모델링할 수 있다. 그리고 쌍선형 시스템은 일반적인 조건하에서 임의의 정밀도를 갖는 어떤 Volterra 시스템도 근사화 할 수 있는 장점을 지니고 있다. 이와 같은 장점으로 인하여 쌍선형 시스템은 많은 분야에서 이용하고 있다[4].

쌍선형 시스템에 대한 안전도에 관한 연구는 Kuburusly와 Liu등에 의하여 연구되어 왔다. Kuburusly[5]는 쌍선형 시스템이 평균 자승 면에서 안정하기 위한 필요 충분조건을 제시하였고, Liu[6]은 일반적인 쌍선형 시간 수열(time series)의 정상 해가 존재하기 위한 충분조건을 제시하였다. 또한 참고문헌[7]에서도 비슷한 결과를 발표하였다. 그러나 이들 방법들은 Kronecker 곱을 사용하고 있기 때문에 행렬의 차수가 커지면 행렬의 고유치를 계산하는 데 많은 계산량을 필요로 하기 때문에 실제 안정도를 조사하는 데 사용하기는 어려운 단점이 있다. 또한 Lee[8]은 출력 오차 방법을 이용한 최소 평균 자승 쌍선형 필터의 안정도 모니터링에 관한 연구를 발표하였다.

본 논문에서는 쌍선형 시스템에 대한 적응 알고리듬과 구조는 최소 자승 방법과 격자 구조를 선택하였다.

순환 최소 자승 알고리듬을 이용하여 직접형으로 유도하는 경우 계산 량이 $O(N^4)$가 되고, 고속 알고리듬을 사용하는 경우 계산 량을 줄일 수 있으나 수치적 문제점(numerical problem)을 지니고 있다. 그래서 수렴 속도가 빠르고 수치적으로 안정한 격자 구조를 이용하였고, 사전 추정 오차와 사후 추정 오차를 기하 평균한 오차를 사용하는 QR 분해법을 이용하여 최소 자승 격자 알고리듬을 유도하였다[9,10,11].

일반적으로 데이터 행렬에 직교 행렬을 이용하는 분해 방법에는 SVD(singular value decomposition) 및 QRD(QR-decomposition) 등이 있고, 이들 중 QRD는 임의의 데이터 행렬의 앞에 직교 행렬을 곱하여 데이터 행렬을 삼각형화 시킨다[9,11].

이와 같이 임의의 데이터 행렬에 대하여 삼각형화 처리를 할 수 있는 기법으로는 Householder 변환 또는 Givens 회전법 등이 있으나, Givens 회전법이 Householder 변환보다 효율적으로 삼각형 처리를 순환적으로 할 수 있어 적응 필터링에 적합하다.

Givens 회전을 이용한 QRD 알고리듬은 Gentleman과 Kung에 의하여 하나의 PE (processing element) 와 이웃하는 PE 끼리 서로 결합하여 VLSI 화하기 매우 용이한 삼각형 어레이 형태를 처음 제안하였다[9,12,13]. 또한 최소 자승 알고리듬을 유도하는 경우 상관행렬의 역행렬을 구해야 하는 데 QR 분해법을 사용하면 역행렬을 구할 필요가 없고 스칼라 처리를 할 수 있는 장점이 있다.

ARMA, Volterra 필터 및 쌍선형 필터는 일반적으로는 입력 데이터들에 대한 시간 천이(time-shift) 성질이 부분적으로 성립한다. 그러므로 부분적으로 성립하는 시간 천이 성질을 이용하면 등가의 다채널 형태로 변환할 수 있다.

시간 천이 성질이 부분적으로 성립하는 경우의 격자 구조에 대한 연구는 Ling 과 Proakis[14]등에 의하여 연구되어 졌다.

비선형 필터에 대한 연구에서 2차 Volterra 필터의 격자 알고리듬이 Mathews에 의해[15,16], 쌍선형 필터의 격자 구조와 격자 알고리듬이 Baik[17,18]에 의해 연구되었으며, 또한 Lee[19]에 의하여 Volterra 필터의 고속 RLS 알고리듬이 유도되었다.

QR 분해법을 이용한 다채널 필터의 격자 알고리듬은 Yang이 기존의 다채널 격자 알고리듬을 Givens 회전 법과 대수학적으로 등가인 Cholesky 분해를 이용하여 QR-MLSL 알고리듬을 제안하였다[20]. 비선형 필터에 대한 연구는 2차 Volterra 필터의 QRD-RLS 알고리듬은 Mathews 와 Syed[15,16]에 의해 제안되었고 참고문헌[18]에 쌍선형 필터의 QRD-RLS 알고리듬이 제안되었다. 이 방법들은 기존의 필터 식에 Givens 회전법과 등가인 Cholesky 분해 방법과 직교 행렬의 성질을 이용하여 유도되었다. 그러나 위와 같은 알고리듬은 완전한 QRD 분해를 이용한 격자 알고리듬이라고는 보기 어렵다.

본 논문에서는 가장 격자 구조와 비슷한 QRD 알고리듬을 얻기 위하여 비선형 필터에 대한 적절한 채널 분해 과정을 제안하였으며 이 과정에서 얻어진 입력 벡터를 이용하여 QR 분해법을 이용한 쌍선형 적응 격자 알고리듬을 유도하였다. 이 분할 과정을 이용하여 쌍선형 필터의 격자식을 쉽게 얻을 수 있고, 또한 개발된 QRD 알고리듬으로부터 직접형 필터의 계수는 기존의 방법을 이용하여 구할 수 있다.

2장에서는 쌍선형 필터의 격자 구조를 개발하기 위한 채널 분해법에 대하여 다루었고, 3장에서는 채널 분해된 입력 벡터를 이용하여 입력 행렬을 QR 분해법을 이용하여 직교화 하는 방법을 제시하였다.

그리고 4장에서는 제안한 알고리듬을 식오차 방법과 출력오차 방법으로 쌍선형 시스템의 식별 문제에 적용하여 제안한 알고리듬의 성능을 평가하였다.

마지막으로 결론을 5장에 나타내었다.

Ⅱ. 채널 분해법을 이용한 비선형 격자 구조

시간 k일 때 적응 필터의 계수에 의한 시간 n에서 쌍선형 모델의 출력 $y(n)$의 추정치를 $\hat{y}_k(n)$라 하면 $\hat{y}_k(n)$는 다음과 같다.

$$\begin{aligned}\hat{y}_k(n) &= \sum_{i=0}^{N-1} a_i(k)x(n-i) + \sum_{i=0}^{N-1}\sum_{j=1}^{N-1} c_{ij}(k)x(n-i)y(n-j) \\ &+ \sum_{j=1}^{N-1} b_j(k)y(n-j) = \boldsymbol{w}^T(k)\,\boldsymbol{u}(n)\end{aligned} \tag{1}$$

여기서

$$\begin{aligned}\boldsymbol{u}(n) = [\ & x(n),\cdots,x(n-N+1),x(n)y(n-1),\cdots,y(n-N+1) \\ & y(n-N+1),y(n-1),\cdots,y(n-N+1)]^T\end{aligned} \tag{2}$$

$$\begin{aligned}\boldsymbol{w}(k) = [\ & a_0(k),\cdots,a_{N-1}(k),c_{0,1}(k),\cdots,c_{N-1,N-1}(k), \\ & b_1(k),\cdots,b_{N-1}(k)]^T\end{aligned} \tag{3}$$

이고, $(\cdot)^T$는 전치 행렬을 나타낸다.

식 (1)과 같은 비선형 모델을 격자 필터로 개발하기 위하여 본 논문에서는 참고문헌[13-14]에서와 같이 비선형 문제를 등가의 다채널 필터링 문제로 변환시켰다.

이 방법은 입력 벡터 $\boldsymbol{u}(n)$을 $2N$ 개의 부분집합으로 나눌 수 있다.

그리고 $2N$ 개의 부분집합은 각각 서로 다른 입력 채널의 연속적인 입력 신호로 이루어졌다고 생각할 수 있다[17].

$$\begin{array}{lllll} CH\,1 & x(n) & x(n-1) & x(n-2) & \cdots\ x(n-N+1) \\ CH\,2 & & y(n-1) & y(n-2) & \cdots\ y(n-N+1) \\ CH\,3 & & x(n)y(n-1) & x(n-1)y(n-2) & \cdots\ x(n-N+2)y(n-N+1) \\ CH\,4 & & x(n-1)y(n-1) & x(n-2)y(n-2) & \cdots\ x(n-N+1)y(n-N+1) \\ CH\,5 & & & x(n-2)y(n-1) & \cdots\ x(n-N+1)y(n-N+2) \\ CH\,6 & & & x(n)y(n-2) & \cdots\ x(n-N+3)y(n-N+1) \\ \vdots & & & \cdots\cdots & \\ CH\,2N-1 & & & & x(n-N+1)y(n-1) \\ CH\,2N & & & & x(n)y(n-N+1) \end{array} \tag{4}$$

위의 식에서 채널 1은 N, 채널 2, 3, 4는 $N-1$ 및 $k\geq 3$일때 채널 $2k-1$와 $2k$는 $N-k+1$개의 지연을 갖는다. 전체 계수의 수는 N^2+N-1이다.

표현을 간단하게 하기 위하여 같은 수의 계수를 갖는 채널들을 하나의 채널 벡터로 묶으면 $2N$개의 채널은 다음과 같이 N개의 채널로 정리할 수 있다.

$$\begin{aligned}\boldsymbol{x}_1(n) &= [\ x(n)]^T \\ \boldsymbol{x}_2(n) &= [\ y(n-1),x(n-1)y(n-1),x(n)y(n-1)]^T \\ \boldsymbol{x}_3(n) &= [\ x(n-2)y(n-1),x(n)y(n-2)]^T \\ &\vdots \\ \boldsymbol{x}_N(n) &= [\ x(n-N+1)y(n-1),x(n)y(n-N+1)]^T\end{aligned} \tag{5}$$

위의 관계식을 이용하여 입력 벡터 $X_N(n)$를 표현하면 아래와 같다.

$$\begin{aligned}X_N(n) = [\ & x_0^T(n),\cdots,x_0^T(n-N+1),\boldsymbol{x}_1^T(n), \\ & \cdots,\boldsymbol{x}_1^T(n-N+2),\cdots\boldsymbol{x}_{N-1}^T(n)]^T\end{aligned} \tag{6}$$

쌍선형 필터의 격자식을 유도하기 위하여 식 (6)으로 정의된 입력 벡터로부터 순방향 및 역방향 예측 오차를 정의해야 한다.

시간 n에서 $m=0$인 경우 이용 가능한 데이터는 $x_0^T(n)$이 된다. 입력 벡터는 다음과 같이 정의 할 수 있다.

$$X_1(n) = [\ x_0^T(n)]^T \tag{7}$$

그러면 순방향 예측오차 $f_0(n)$과 역방향 예측오차 $b_0(n)$을 다음과 같이 정의 할 수 있다.

$$f_0(n) = x_0(n) = x_{0\cdot 0}^f(n) = T_1X_1(n) \tag{8}$$

$$b_0(n) = x_0(n) = x_0^b(n) = S_1X_1(n) \tag{9}$$

다음으로 시간 n에서 $m=1$인 경우 이용 가능한 데이터는 $x_0^T(n)$, $x_0^T(n-1)$과 $\boldsymbol{x}_1^T(n)$이 된다.

$$X_2(n) = [\ x_0^T(n),x_0^T(n-1),\boldsymbol{x}_1^T(n)]^T \tag{10}$$

그러면 순방향 및 역방향 예측 오차 벡터 $f_1(n)$과 $\boldsymbol{b}_1(n)$은 각각 다음과 같이 정의 할 수 있다.

$$\begin{aligned}f_1(n) &= \begin{bmatrix} x_0(n) \\ \boldsymbol{x}_1(n)\end{bmatrix} - A_1^T(n)[\ x_0(n-1)] \\ &= x_{1\cdot 1}^f(n) - A_1^T(n)X_1(n-1) \\ &= [\ I, -A_1^T(n)]\ T_2X_2(n)\end{aligned} \tag{11}$$

$$\begin{aligned} \boldsymbol{b}_1(n) &= \begin{bmatrix} x_0(n-1) \\ \boldsymbol{x}_1(n) \end{bmatrix} - B_1^T(n)[\ x_0(n)] \\ &= \boldsymbol{x}_1^b(n) - B_1^T(n)X_1(n) \\ &= [\ -B_1^T(n), I]\ S_2X_2(n) \end{aligned} \tag{12}$$

여기서 $\boldsymbol{x}_1(n)$은 $m=1$에서 채널에 처음으로 나타내는 입력벡터이고 $A_1(n)$과 $B_1(n)$는 순방향 및 역방향 예측 오차 필터의 계수 벡터이다. 그리고 T_2과 S_2는 변환행렬이다.

이와 같은 방법을 적용하여 차수 m인 경우를 고려하면 다음과 같다.

$$\begin{aligned} \boldsymbol{f}_m(n) &= \boldsymbol{x}_{m:m}^f(n) - A_m^T(n)X_m(n-1) \\ &= [\ I, -A_m^T(n)]\ T_{m+1}X_{m+1}(n) \end{aligned} \tag{13}$$

$$\begin{aligned} \boldsymbol{b}_m(n) &= \boldsymbol{x}_m^b(n) - B_m^T(n)X_m(n) \\ &= [\ -B_m^T(n), I]\ S_{m+1}X_{m+1}(n) \end{aligned} \tag{14}$$

여기서 $A_m(n)$과 $B_m(n)$는 순방향 및 역방향 예측 오차 필터의 계수이고, $\boldsymbol{x}_{m:m}^f(n)$과 $\boldsymbol{x}_m^b(n)$은 순방향 및 역방향 예측기의 원하는 응답(desired response)의 역할을 한다. 또한 변환행렬 T_{m+1}과 S_{m+1}은 입력 벡터를 $X_{m+1}(n)$을 다음과 같이 분할한다.

$$T_{m+1}X_{m+1}(n) = \begin{bmatrix} \boldsymbol{x}_{m:m}^f(n) \\ X_m(n-1) \end{bmatrix} \tag{15}$$

$$S_{m+1}X_{m+1}(n) = \begin{bmatrix} X_m(n) \\ \boldsymbol{x}_m^b(n) \end{bmatrix} \tag{16}$$

$\boldsymbol{f}_m(n)$은 $\boldsymbol{x}_{m:m}^f(n)$을 $X_m(n-1)$로 최소 자승 예측할 때 발생하는 오차이고 $\boldsymbol{b}_m(n)$은 $\boldsymbol{x}_m^b(n)$을 $X_m(n)$으로 최소 자승 예측할 때 발생되는 오차를 의미한다.

즉, $\boldsymbol{f}_m(n)$은 $\boldsymbol{x}_{m:m}^f(n)$을 $X_m(n-1)$으로 직교 투영시킨 결과이고, $\boldsymbol{b}_m(n)$은 $\boldsymbol{x}_m^b(n)$을 $X_m(n)$에 직교 투영시킨 결과를 나타낸다.

식 (15)에서 $X_{m-1}(n-1)$을 S_{m-1}을 이용하여 재 분할하면 역방향 예측오차에 대한 정보를 얻을 수 있고 또한 식 (16)에서 $X_m(n)$을 T_m을 이용하여 분할하면 순방향에 대한 정보를 얻을 수 있다.

이러한 과정을 간단히 표현하면 다음과 같이 쓸 수 있다.

$$\begin{aligned} \begin{bmatrix} I & 0 \\ 0 & S_m \end{bmatrix} \{T_{m+1}X_{m+1}(n)\} &= \begin{bmatrix} I & 0 \\ 0 & S_m \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{m:m}^f(n) \\ X_m(n-1) \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{x}_{m:m}^f(n) \\ \begin{bmatrix} X_{m-1}(n-1) \\ \boldsymbol{x}_{m-1}^b(n-1) \end{bmatrix} \end{bmatrix} \end{aligned} \tag{17}$$

$$\begin{aligned} \begin{bmatrix} T_m & 0 \\ 0 & 1 \end{bmatrix} \{S_{m+1}X_{m+1}(n)\} &= \begin{bmatrix} T_m & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_m(n) \\ \boldsymbol{x}_m^b(n) \end{bmatrix} \\ &= \begin{bmatrix} \begin{bmatrix} \boldsymbol{x}_{m-1:m-1}^f(n) \\ X_{m-1}(n-1) \end{bmatrix} \\ \boldsymbol{x}_m^b(n) \end{bmatrix} \end{aligned} \tag{18}$$

또한 $\boldsymbol{x}_{m:m}^f(n)$과 $\boldsymbol{x}_m^b(n)$는 다음과 같은 성질이 성립한다.

$$\boldsymbol{x}_{m:m}^f(n) = \begin{bmatrix} \boldsymbol{x}_{m-1:m-1}^f(n) \\ \boldsymbol{x}_m(n) \end{bmatrix} \tag{19}$$

$$\boldsymbol{x}_m^b(n) = \begin{bmatrix} \boldsymbol{x}_{m-1}^b(n-1) \\ \boldsymbol{x}_m(n) \end{bmatrix} \tag{20}$$

위의 결과를 이용하면 다음과 같이 순방향 입력 벡터 $X_{m+1}^f(n)$과 $X_{m+1}^b(n)$를 정의 할 수 있다.

$$\begin{aligned} X_{m+1}^{fT}(n) = [\ &\boldsymbol{x}_{m-1:m-1}^{fT}(n), \boldsymbol{x}_m^T(n), X_{m-1}^T(n-1), \\ &\boldsymbol{x}_{m-1}^{bT}(n-1)\ ] \end{aligned} \tag{21}$$

$$\begin{aligned} X_{m+1}^{bT}(n) = [\ &\boldsymbol{x}_{m-1:m-1}^{fT}(n), X_{m-1}^T(n-1), \\ &\boldsymbol{x}_{m-1}^{bT}(n-1), \boldsymbol{x}_m^T(n)\ ] \end{aligned} \tag{22}$$

위의 두 식은 구성 요소는 서로 같고 단지 배열만 다르게 구성되어 있는 특징이 있다. 식 (21)에서 $\boldsymbol{x}_{m-1:m-1}^f(n)$을 $X_{m-1}(n)$에 직교투영 시키면 $\boldsymbol{f}_{m-1}(n)$을 얻을 수 있고, $\boldsymbol{f}_{m-1}^{(m)}(n)$은 $\boldsymbol{x}_m(n)$을 $X_{m-1}(n)$에 직교투영 시켜 얻는다. 그리고 $\boldsymbol{f}_m(n)$ 은 $\boldsymbol{x}_{m-1:m-1}^f(n)$을 $X_{m-1}(n)$과 $\boldsymbol{x}_{m-1}^b(n-1)$에 직교투영 시키면 얻을 수 있고, $\boldsymbol{f}_m^{(m)}(n)$은 $\boldsymbol{x}_m(n)$을 $X_{m-1}(n)$과 $\boldsymbol{x}_{m-1}^b(n-1)$에 직교투영 시켜 얻는다.

$X_{m+1}^f(n)$을 이용하여 순방향 예측 오차 $\boldsymbol{f}_m(n)$, 순방향 보조 예측 오차 $\boldsymbol{f}_m^{(m)}(n)$을 각각 구할 수 있다. 본 논문에서는 $[\ X_{m-1}(n-1)\ ]$과 $[\ X_{m-1}(n-1), \boldsymbol{x}_{m-1}^b(n-1)\ ]$의 각각의 요소를 직교화 시키는 과정을 QR 분해법을 이용하여 수행하였다.

그리고 역방향 예측 오차 벡터에서도 위와 비슷한 과정을 통하여 얻어진다.

m차 순방향 예측 오차 $\boldsymbol{f}_m(n)$은 다음과 같이 구해진다.

$$\bar{\boldsymbol{f}}_m(n) = \begin{bmatrix} \boldsymbol{f}_m(n) \\ \boldsymbol{f}_m^{(m)}(n) \end{bmatrix} \tag{23}$$

비슷한 방법으로 $X_{m+1}^b(n)$을 이용하여 순방향 예측 오차 $\boldsymbol{f}_{m-1}(n)$, 역방향 보조 예측 오차 $\boldsymbol{b}_m^{(m)}(n)$ 및 역방

향 예측 오차 $\boldsymbol{b}_{m-1}(n-1)$을 각각 구할 수 있다. m차 역방향 예측 오차 $\boldsymbol{b}_m(n)$은 다음과 같이 구해진다.

$$\bar{\boldsymbol{b}}_m(n)=\begin{bmatrix}\boldsymbol{b}_m(n)\\ \boldsymbol{b}_m^{(m)}(n)\end{bmatrix} \tag{24}$$

위의 방법은 이용하여 격자식을 유도하면 참고문헌[17]에서 유도한 격자식과 같아지게 된다. 그리고 반사계수가 순방향과 역방향이 다른 경우 본 논문에서 제안한 방법을 이용하면 격자식을 쉽게 얻을 수 있다.

Ⅲ. QR 분해법을 이용한 쌍선형 순환 최소 자승 격자 알고리듬

본 절에서는 QR 분해법을 이용하여 순환 최소 자승 쌍선형 격자 알고리듬을 개발하겠다. QR분해 방법은 참고문헌[9,11]에 매우 자세히 언급하고 있고, 본 논문은 참고문헌[11]의 방법을 적용하여 유도하였다.

식 (19)을 이용하여 순방향 데이터 행렬 $D_{m+1}^f(n)$을 구성하면 다음과 같다.

$$\begin{aligned}D_{m+1}^f(n) &= \begin{bmatrix} \boldsymbol{x}_{m-1,m-1}^{fT}(1) & \boldsymbol{x}_m^T(1) & X_{m-1}^T(0) & \boldsymbol{x}_{m-1}^{bT}(0)\\ \boldsymbol{x}_{m-1,m-1}^{fT}(2) & \boldsymbol{x}_m^T(2) & X_{m-1}^T(1) & \boldsymbol{x}_{m-1}^{bT}(1)\\ \boldsymbol{x}_{m-1,m-1}^{fT}(3) & \boldsymbol{x}_m^T(3) & X_{m-1}^T(2) & \boldsymbol{x}_{m-1}^{bT}(2)\\ \vdots & \vdots & \vdots & \vdots\\ \boldsymbol{x}_{m-1,m-1}^{fT}(n-1) & \boldsymbol{x}_m^T(n-1) & X_{m-1}^T(n-2) & \boldsymbol{x}_{m-1}^{bT}(n-2)\\ \boldsymbol{x}_{m-1,m-1}^{fT}(n) & \boldsymbol{x}_m^T(n) & X_{m-1}^T(n-1) & \boldsymbol{x}_{m-1}^{bT}(n-1)\end{bmatrix}\\ &= \begin{bmatrix}\boldsymbol{d}_{m-1}^f(n) & \boldsymbol{d}_m^{f(m)}(n) & D_{m-1}(n-1) & \boldsymbol{d}_{m-1}^b(n-1)\end{bmatrix}\end{aligned} \tag{25}$$

순방향 예측 오차에 대한 QR 분해법을 이용한 최소 자승 알고리듬을 계산하기 위하여 $D_{m-1}(n-1)$을 삼각화 시키기 위하여 회전 행렬 $Q_m(n-1)$을 정의해야 된다.

또한 사후 예측 오차를 계산하기 위한 각 변수를 계산하기 위하여 위의 식을 다음과 같이 변화시킬 수 있다.

$$M_1=\begin{bmatrix}\boldsymbol{d}_{m-1}^f(n) & \boldsymbol{d}_m^{f(m)} & \boldsymbol{d}_{m-1}(n-1) & \boldsymbol{d}_{m-1}^b(n-1)\\ \boldsymbol{y}(n-1) & \boldsymbol{\pi}_{m-1} & & \end{bmatrix} \tag{26}$$

여기서 $\boldsymbol{y}(n-1)$은 원하는 응답 벡터이고 $\boldsymbol{\pi}_{m-1}$은 각각 다음과 같다.

$$\boldsymbol{y}(n-1)=[\ y(0),y(1),\cdots,y(n-1)]^T \tag{27}$$

$$\boldsymbol{\pi}_{m-1}=[\ 0,0,\cdots,1]^T \tag{28}$$

식 (26)에서 $D_{m-1}(n-1)$의 모든 요소는 직교 회전 행렬 $Q_m(n-1)$에 의하여 이루어지고 $D_{m-1}(n-1)$은 상관행렬을 삼각행렬로 분해된 형태로 나타나게 된다. $D_{m-1}(n-1)$을 직교화 시키기 위하여 M_1에 직교 회전 행렬 $Q_m(n-1)$을 적용하면 QR 분해의 정의에 의하여 다음과 같이 쓸 수 있다.

$$\begin{aligned}&Q_m(n-1)M_1=\\ &\begin{bmatrix}\boldsymbol{p}_{m-1}^f(n) & \boldsymbol{p}_{m-1}^{f(m)}(n) & R_{m-1}(n-1) & \boldsymbol{p}_{m-1}^b(n-1) & \boldsymbol{p}_{m-1}(n-1) & \boldsymbol{q}_{m-1}(n-1)\\ \boldsymbol{v}_{m-1}^f(n) & \boldsymbol{v}_{m-1}^{f(m)}(n) & O & \boldsymbol{v}_{m-1}^b(n-1) & \boldsymbol{v}_{m-1}(n-1) & \boldsymbol{g}_{m-1}(n-1)\end{bmatrix}\\ &=M_2\end{aligned} \tag{29}$$

여기서

$$\boldsymbol{g}_{m-1}(n-1)=[\ \boldsymbol{0}^T,\tilde{\alpha}_{m-1}(n-1)]^T \tag{30}$$

QR 분해법의 정의에서, $\boldsymbol{v}_{m-1}^f(n)$, $\boldsymbol{v}_{m-1}^{(m)}(n)$, $\boldsymbol{v}_{m-1}^b(n-1)$ 및 $\boldsymbol{v}_{m-1}(n-1)$은 시간 갱신(time update)을 하는 벡터들 이므로 이들에서 시간 n일때의 항들을 분리하여 나타내면 다음과 같다.

$$\begin{aligned}&M_2=\\ &\begin{bmatrix}\boldsymbol{p}_{m-1}^f(n) & \boldsymbol{p}_{m-1}^{f(m)}(n) & R_{m-1}(n-1) & \boldsymbol{p}_{m-1}^b(n-1) & \boldsymbol{p}_{m-1}(n-1) & \boldsymbol{q}_{m-1}(n-1)\\ \lambda^{1/2}\boldsymbol{v}_{m-1}^f(n-1) & \lambda^{1/2}\boldsymbol{v}_{m-1}^{f(m)}(n-1) & O & \lambda^{1/2}\boldsymbol{v}_{m-1}^b(n-2) & \lambda^{1/2}\boldsymbol{v}_{m-1}(n-2) & 0\\ \tilde{\boldsymbol{f}}_{m-1}^T(n) & \tilde{\boldsymbol{f}}_{m-1}^{(m)T}(n) & \boldsymbol{0}^T & \tilde{\boldsymbol{b}}_{m-1}^T(n-1) & \tilde{e}_{m-1}(n-1) & \tilde{\alpha}_{m-1}(n-1)\end{bmatrix}\end{aligned} \tag{31}$$

여기서 $\tilde{\boldsymbol{f}}_{m-1}(n)$과 $\tilde{\boldsymbol{f}}_{m-1}^{(m)}(n)$은 각 정규화된 순방향 및 순방향 보조 예측 오차 벡터이다. $\tilde{\boldsymbol{b}}_{m-1}(n-1)$은 각 정규화된 역방향 예측 오차 벡터이다. $\tilde{e}_{m-1}(n-1)$은 각 정규화된 예측 오차 벡터이다. $\tilde{\alpha}_{m-1}(n-1)$은 각 변수이다.

직교 행렬 $Q_m^f(n-1)$이 이미 계산되어 있다고 가정하면, $\boldsymbol{v}_{m-1}^b(n-2)$을 시간 $n-2$부터 위쪽으로 직교화 시키면 차수 갱신된 $R_m(n-1)$을 얻을 수 있다.

$$R_m(n-1)=\begin{bmatrix}R_{m-1}(n-1) & \boldsymbol{p}_{m-1}^b(n-1)\\ & \lambda^{1/2}\bar{R}_{m-1}^{b/2}(n-2)\end{bmatrix} \tag{32}$$

식 (32)을 식 (31)에 대입하면 다음의 식을 얻을 수 있다.

$$\begin{aligned}&\begin{bmatrix}Q_m^f(n-1) & 0\\ 0^T & 1\end{bmatrix}M_2=\\ &\begin{bmatrix}\boldsymbol{p}_{m-1}^f(n) & \boldsymbol{p}_{m-1}^{f(m)}(n) & R_{m-1}(n-1) & \boldsymbol{p}_{m-1}^b(n-1) & \boldsymbol{p}_{m-1}(n-1) & \boldsymbol{q}_{m-1}(n-1)\\ \lambda^{1/2}\boldsymbol{\beta}_{m-1}^f(n-1) & \lambda^{1/2}\boldsymbol{\beta}_{m-1}^{f(m)}(n-1) & 0 & \lambda^{1/2}R_{m-1}^{b/2}(n-2) & \lambda^{1/2}\boldsymbol{\beta}_{m-1}(n-2) & 0\\ \lambda^{1/2}\boldsymbol{\varphi}_{m-1}^f(n-1) & \lambda^{1/2}\boldsymbol{\varphi}_{m-1}^{f(m)}(n-1) & O & 0 & \lambda^{1/2}\boldsymbol{\varphi}_{m-1}(n-2) & 0\\ \tilde{\boldsymbol{f}}_{m-1}^T(n) & \tilde{\boldsymbol{f}}_{m-1}^{(m)T}(n) & 0^T & \tilde{\boldsymbol{b}}_{m-1}^T(n-1) & e_{m-1}(n-1) & \alpha_{m-1}(n-1)\end{bmatrix}\\ &=M_3\end{aligned} \tag{33}$$

여기서 $\bar{R}_{m-1}^{b/2}(n-2)$는 시간 $n-2$에서 제곱된 $m-1$차인 역방향 예측 에너지(backward prediction energy)를 나타낸다. 식 (32)을 완전히 직교화 시키기 위해서는 $\tilde{\boldsymbol{b}}_{m-1}(n-1)$을 직교화 시켜야 한다. $\tilde{\boldsymbol{b}}_{m-1}(n-1)$을 Givens 회전시키기 위하여 식 (33)에 $\bar{Q}_m^f$

$(n-1)$을 적용시키면 다음과 같은 식을 얻을 수 있다.

$$\begin{bmatrix} \boldsymbol{p}^f_{m-1}(n) & \boldsymbol{p}^{f,m)}_{m-1}(n) & R_{m-1}(n-1) & \boldsymbol{p}^b_{m-1}(n-1) & \boldsymbol{p}_{m-1}(n-1) & \boldsymbol{q}_{m-1}(n-1) \\ \tilde{\boldsymbol{\beta}}^f_{m-1}(n) & \tilde{\boldsymbol{\beta}}^{f,m)}_{m-1}(n) & 0 & \tilde{R}^{b/2}_{m-1}(n-1) & \tilde{\boldsymbol{\beta}}_{m-1}(n-1) & \bar{\boldsymbol{q}}_m(n-1) \\ \lambda^{1/2}\boldsymbol{\varphi}^f_{m-1}(n-1) & \lambda^{1/2}\boldsymbol{\varphi}^{f,m)}_{m-1}(n-1) & O & 0 & \lambda^{1/2}\boldsymbol{\varphi}_{m-1}(n-2) & 0 \\ \tilde{\boldsymbol{f}}^T_m(n) & \tilde{\boldsymbol{f}}^{(m)T}_m(n) & 0^T & 0^T & \tilde{e}_m(n-1) & \tilde{\alpha}_m(n-1) \end{bmatrix} \tag{34}$$

식 (34)에서 차수 갱신된 완벽한 $R_m(n-1)$을 얻을 수 있다.

$$R_m(n-1)=\begin{bmatrix} R_{m-1}(n-1) & \boldsymbol{p}^b_{m-1}(n-1) \\ & \tilde{R}^{b/2}_{m-1}(n-1) \end{bmatrix} \tag{35}$$

식 (33)와 식 (34)에서 각 정규 화된 순방향 예측 오차(angle normalized forward prediction error)에 대한 블록 형태의 알고리듬은 다음과 같이 된다.

$$\begin{aligned} &\bar{Q}^f_m(n-1)\times \\ &\begin{bmatrix} \lambda^{1/2}\tilde{\boldsymbol{\beta}}^f_{m-1}(n-1) & \lambda^{1/2}\tilde{\boldsymbol{\beta}}^{f,m)}_{m-1}(n-1) & \lambda^{1/2}\tilde{R}^{b/2}_{m-1}(n-2) & \lambda^{1/2}\tilde{\boldsymbol{\beta}}_{m-1}(n-2) & 0 \\ \tilde{\boldsymbol{f}}^T_{m-1}(n) & \tilde{\boldsymbol{f}}^{(m)T}_{m-1}(n) & \tilde{\boldsymbol{b}}^T_{m-1}(n-1) & \tilde{e}_{m-1}(n-1) & \tilde{\alpha}_{m-1}(n-1) \end{bmatrix} \\ &=\begin{bmatrix} \tilde{\boldsymbol{\beta}}^f_{m-1}(n) & \tilde{\boldsymbol{\beta}}^{f,m)}_{m-1}(n) & \tilde{R}^{b/2}_{m-1}(n-1) & \tilde{\boldsymbol{\beta}}_{m-1}(n-1) & \bar{\boldsymbol{q}}_m(n-1) \\ \tilde{\boldsymbol{f}}^T_m(n) & \tilde{\boldsymbol{f}}^{(m)T}_m(n) & 0^T & \tilde{e}_m(n-1) & \tilde{\alpha}_m(n-1) \end{bmatrix} \end{aligned} \tag{36}$$

여기서 $\bar{Q}^f_m(n-1)$의 형태는 다음과 같다.

$$\bar{Q}^f_m(n-1)=\begin{bmatrix} \boldsymbol{c}^f_{m-1}(n-1) & 0^T & \boldsymbol{s}^f_{m-1}(n-1) \\ 0 & I & 0 \\ -\boldsymbol{s}^f_{m-1}(n-1) & 0^T & \boldsymbol{c}^f_{m-1}(n-1) \end{bmatrix} \tag{37}$$

그러면 각 정규화된 역방향 예측 에너지, 순방향 예측 오차, 순방향 보조 예측 오차, 접합 과정 추정 오차 및 각 변수를 구하기 위한 순환 식은 다음과 같다.

$$\tilde{R}^b_{m-1}(n-1)=\lambda\tilde{R}^b_{m-1}(n-2)+[\ \tilde{\boldsymbol{b}}^T_{m-1}(n-1)]^{\ 2} \tag{38}$$

$$\boldsymbol{c}^f_{m-1}(n-1)=\lambda^{1/2}\tilde{R}^{b/2}_{m-1}(n-2)\,\tilde{R}^{-b/2}_{m-1}(n-1) \tag{39}$$

$$\boldsymbol{s}^f_{m-1}(n-1)=\tilde{\boldsymbol{b}}^T_{m-1}(n-2)\,\tilde{R}^{-b/2}_{m-1}(n-1) \tag{40}$$

$$\begin{aligned} \tilde{\boldsymbol{\beta}}^f_{m-1}(n) &= \boldsymbol{c}^f_{m-1}(n-1)\lambda^{1/2}\tilde{\boldsymbol{\beta}}^f_{m-1}(n-1) \\ &\quad +\boldsymbol{s}^f_{m-1}(n-1)\,\tilde{\boldsymbol{f}}^T_{m-1}(n) \end{aligned} \tag{41}$$

$$\begin{aligned} \tilde{\boldsymbol{f}}^f_m(n) &= \boldsymbol{c}^f_{m-1}(n-1)\,\tilde{\boldsymbol{f}}^T_{m-1}(n) \\ &\quad -\boldsymbol{s}^f_{m-1}(n-1)\lambda^{1/2}\tilde{\boldsymbol{\beta}}^f_{m-1}(n-1) \end{aligned} \tag{42}$$

$$\begin{aligned} \tilde{\boldsymbol{\beta}}^{f,m)}_{m-1}(n) &= \boldsymbol{c}^f_{m-1}(n-1)\lambda^{1/2}\tilde{\boldsymbol{\beta}}^{f,m)}_{m-1}(n-1) \\ &\quad +\boldsymbol{s}^f_{m-1}(n-1)\,\tilde{\boldsymbol{f}}^{(m)T}_{m-1}(n) \end{aligned} \tag{43}$$

$$\begin{aligned} \tilde{\boldsymbol{f}}^{f,m)}_m(n) &= \boldsymbol{c}^f_{m-1}(n-1)\,\tilde{\boldsymbol{f}}^{(m)T}_{m-1}(n)-\boldsymbol{s}^f_{m-1}(n-1)\lambda^{1/2} \\ &\quad \tilde{\boldsymbol{\beta}}^{f,m)}_{m-1}(n-1) \end{aligned} \tag{44}$$

$$\begin{aligned} \tilde{\boldsymbol{\beta}}_{m-1}(n-1) &= \boldsymbol{c}^f_{m-1}(n-1)\lambda^{1/2}\tilde{\boldsymbol{\beta}}_{m-1}(n-2) \\ &\quad +\boldsymbol{s}^f_{m-1}(n-1)\,\tilde{e}_{m-1}(n-1) \end{aligned} \tag{45}$$

$$\begin{aligned} \tilde{e}_m(n-1) &= \boldsymbol{c}^f_{m-1}(n-1)\,\tilde{e}_{m-1}(n-1) \\ &\quad -\boldsymbol{s}^f_{m-1}(n-1)\lambda^{1/2}\tilde{\boldsymbol{\beta}}_{m-1}(n-2) \end{aligned} \tag{46}$$

$$\bar{\boldsymbol{q}}_m(n-1)=\tilde{\boldsymbol{s}}^f_{m-1}(n-1)\,\tilde{\alpha}_{m-1}(n-1) \tag{47}$$

$$\tilde{\alpha}_m(n-1)=\tilde{\boldsymbol{c}}^f_{m-1}(n-1)\,\tilde{\alpha}_{m-1}(n-1) \tag{48}$$

식 (44)을 이용하여 역방향 데이터 행렬 $D^b_{m+1}(n)$을 구성하면 다음과 같다.

$$\begin{aligned} D^b_{m+1}(n) &= \begin{bmatrix} \boldsymbol{x}^{fT}_{m-1,m-1}(1) & X^T_{m-1}(0) & \boldsymbol{x}^{bT}_{m-1}(0) & \boldsymbol{x}^T_m(1) \\ \boldsymbol{x}^{fT}_{m-1,m-1}(2) & X^T_{m-1}(1) & \boldsymbol{x}^{bT}_{m-1}(1) & \boldsymbol{x}^T_m(2) \\ \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{x}^{fT}_{m-1,m-1}(n) & X^T_{m-1}(n-1) & \boldsymbol{x}^{bT}_{m-1}(n-1) & \boldsymbol{x}^T_m(n) \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{d}^f_{m-1}(n) & D_{m-1}(n-1) & \boldsymbol{d}^b_{m-1}(n-1) & \boldsymbol{d}^{f,m)}_m(n) \end{bmatrix} \end{aligned} \tag{49}$$

사후 예측 오차를 계산하기 위하여 위의 식을 다음과 같이 변화시키자.

$$M_4=\begin{bmatrix} \lambda^{(n-1)/2}\boldsymbol{x}^T_{m-1,m-1}(1) & 0^T & 0^T & \lambda^{(n-1)/2}\boldsymbol{x}^T_m(1) & 0 \\ \boldsymbol{d}^f_{m-1}(n) & D_{m-1}(n-1) & \boldsymbol{d}^b_{m-1}(n-1) & \boldsymbol{d}^{f,m)}_m(n) & \boldsymbol{x}_{m-1} \end{bmatrix} \tag{50}$$

식 (50)의 $D_{m-1}(n-1)$을 Givens 회전시키기 위하여 식 (29)에서 사용한 $Q_m(n-1)$을 적용하면 다음과 같은 식을 얻을 수 있다.

$$\begin{aligned} &\begin{bmatrix} 1 & 0^T \\ 0 & Q_m(n-1) \end{bmatrix} M_4= \\ &\begin{bmatrix} \lambda^{(n-1)/2}\boldsymbol{x}^T_{m-1,m-1}(1) & 0^T & 0^T & \lambda^{(n-1)/2}\boldsymbol{x}^T_m(1) & 0 \\ \boldsymbol{p}^f_{m-1}(n) & R_{m-1}(n-1) & \boldsymbol{p}^b_{m-1}(n-1) & \boldsymbol{p}^{f,m)}_{m-1}(n) & \boldsymbol{q}_{m-1}(n-1) \\ \boldsymbol{v}^f_{m-1}(n) & O & \boldsymbol{v}^b_{m-1}(n-1) & \boldsymbol{v}^{f,m)}_{m-1}(n) & \boldsymbol{g}_{m-1}(n-1) \end{bmatrix} \\ &=M_5 \end{aligned} \tag{51}$$

위의 식에서 $\boldsymbol{v}^f_{m-1}(n)$, $\boldsymbol{v}^b_{m-1}(n-1)$ 및 $\boldsymbol{v}^{f,m)}_{m-1}(n)$은 시간 순환을 하므로 마지막 요소를 식 (31)과 같이 분리하여 나타낼 수 있다.

$$\begin{aligned} M_5= &\begin{bmatrix} \lambda^{(n-1)/2}\boldsymbol{x}^T_{m-1,m-1}(1) & 0^T & 0^T & \lambda^{(n-1)/2}\boldsymbol{x}^T_m(1) & 0 \\ \boldsymbol{p}^f_{m-1}(n) & R_{m-1}(n-1) & \boldsymbol{p}^b_{m-1}(n-1) & \boldsymbol{p}^{f,m)}_{m-1}(n) & \boldsymbol{q}_{m-1}(n-1) \\ \lambda^{1/2}\boldsymbol{v}^f_{m-1}(n-1) & O & \lambda^{1/2}\boldsymbol{v}^b_{m-1}(n-2) & \lambda^{1/2}\boldsymbol{v}^{f,m)}_{m-1}(n-1) & 0 \\ \tilde{\boldsymbol{f}}^T_{m-1}(n) & O & \tilde{\boldsymbol{b}}^T_{m-1}(n-1) & \tilde{\boldsymbol{f}}^{(m)T}_{m-1}(n) & \tilde{\alpha}_{m-1}(n-1) \end{bmatrix} \end{aligned} \tag{52}$$

위의 식에 $\boldsymbol{v}^f_{m-1}(n-1)$을 Givens 회전시키기 위하여 직교 행렬 $Q^b_m(n-1)$을 적용하면 다음의 식을 구할 수 있다.

$$\begin{aligned} &\begin{bmatrix} Q^b_m(n-1) & 0 \\ 0^T & 1 \end{bmatrix} M_5= \\ &\begin{bmatrix} \lambda^{1/2}\tilde{R}^{b/2}_{m-1}(n-1) & 0^T & \lambda^{1/2}\tilde{\boldsymbol{\beta}}^b_{m-1}(n-1) & \lambda^{1/2}\tilde{\boldsymbol{\beta}}^{b,m)}_{m-1}(n-1) & 0 \\ \boldsymbol{p}^f_{m-1}(n) & R_{m-1}(n-1) & \boldsymbol{p}^b_{m-1}(n-1) & \boldsymbol{p}^{f,m)}_{m-1}(n) & \boldsymbol{q}_{m-1}(n-1) \\ 0 & O & \lambda^{1/2}\boldsymbol{\varphi}^b_{m-1}(n-2) & \lambda^{1/2}\boldsymbol{\varphi}^{b,m)}_{m-1}(n-1) & 0 \\ \tilde{\boldsymbol{f}}^T_{m-1}(n) & 0^T & \tilde{\boldsymbol{b}}^T_{m-1}(n-1) & \tilde{\boldsymbol{f}}^{(m)T}_{m-1}(n) & \tilde{\alpha}_{m-1}(n-1) \end{bmatrix} \\ &=M_6 \end{aligned} \tag{53}$$

위의 식의 $\tilde{\boldsymbol{f}}_{m-1}(n)$을 제거하기 위하여 직교 행렬 $\bar{Q}^b_m(n)$을 위의 식에 적용하자.

$$\tilde{Q}_m^b(n)M_6=\begin{bmatrix} \tilde{R}_{m-1}^{1/2}(n) & 0^T & \tilde{\beta}_{m-1}^b(n) & \tilde{\beta}_{m-1}^{b(m)}(n) & \tilde{q}_m(n) \\ \tilde{p}_{m-1}^f(n) & R_{m-1}(n-1) & \tilde{p}_{m-1}^b(n-1) & \tilde{p}_{m-1}^{f(m)}(n) & q_{m-1}(n-1) \\ 0 & O & \lambda^{1/2}\tilde{\varphi}_{m-1}^b(n-2) & \lambda^{1/2}\tilde{\varphi}_{m-1}^{b(m)}(n-1) & 0 \\ 0 & 0^T & \tilde{b}_{m-1}^T(n) & \tilde{b}_m^{(m)T}(n) & \tilde{\alpha}_{m-1}(n) \end{bmatrix} \tag{54}$$

식 (53)과 식 (54)에서 각 정규 화된 역방향 예측 오차(angle normalized backward prediction error)에 대한 블록 형태의 알고리듬은 다음과 같이 된다.

$$\begin{bmatrix} \lambda^{1/2}\tilde{R}_{m-1}^{f/2}(n-1) & \lambda^{1/2}\tilde{\beta}_{m-1}^b(n-1) & \lambda^{1/2}\tilde{\beta}_{m-1}^{b(m)}(n-1) & 0 \\ \tilde{f}_{m-1}^T(n) & \tilde{b}_{m-1}^T(n-1) & \tilde{f}_{m-1}^{(m)T}(n) & \tilde{\alpha}_{m-1}(n-1) \end{bmatrix} \tilde{Q}_m^b(n) \times = \begin{bmatrix} \tilde{R}_{m-1}^{f/2}(n) & \tilde{\beta}_{m-1}^b(n) & \tilde{\beta}_{m-1}^{b(m)}(n) & \tilde{q}_m(n) \\ 0 & \tilde{b}_m^T(n) & \tilde{b}_m^{(m)T}(n) & \tilde{\alpha}_m(n) \end{bmatrix} \tag{55}$$

여기서 $\tilde{Q}_m^b(n)$의 형태는 다음과 같다. $\tilde{R}_{m-1}^{f/2}(n-1)$는 시간 $n-1$에서 제곱된 $m-1$차인 순방향 예측 에너지(backward prediction energy)를 나타낸다.

$$\tilde{Q}_m^b(n)=\begin{bmatrix} c_{m-1}^b(n) & 0^T & s_{m-1}^b(n) \\ 0 & I & 0 \\ -s_{m-1}^b(n) & 0^T & c_{m-1}^b(n) \end{bmatrix} \tag{56}$$

여기서 $c_{m-1}^b(n)$· $s_{m-1}^b(n)$는 각 정규화된 $\tilde{f}_{m-1}^T(n)$을 직교 회전시키기 위한 cosine 및 sine 파라미터들이다.

그러면 각 정규화된 순방향 예측 에너지, 역방향 예측 오차, 역방향 보조 예측 오차 및 각 변수를 구할 수 있는 순환 식은 다음과 같다.

$$\tilde{R}_{m-1}^f(n)=\lambda\tilde{R}_{m-1}^f(n-1)+[\ \tilde{f}_{m-1}^T(n)]^2 \tag{57}$$

$$c_{m-1}^b(n)=\lambda^{1/2}\tilde{R}_{m-1}^{f/2}(n-1)\tilde{R}_{m-1}^{-f/2}(n) \tag{58}$$

$$s_{m-1}^b(n)=\tilde{f}_{m-1}^T(n)\tilde{R}_{m-1}^{-f/2}(n) \tag{59}$$

$$\tilde{\beta}_{m-1}^b(n) = c_{m-1}^b(n)\lambda^{1/2}\tilde{\beta}_{m-1}^b(n-1) + s_{m-1}^b(n)\tilde{b}_{m-1}^T(n-1) \tag{60}$$

$$\tilde{b}_m^T(n) = c_{m-1}^b(n)\tilde{b}_{m-1}^T(n-1)-s_{m-1}^b(n)\lambda^{1/2}\tilde{\beta}_{m-1}^b(n-1) \tag{61}$$

$$\tilde{\beta}_{m-1}^{b(m)}(n) = c_{m-1}^{b(m)}(n)\lambda^{1/2}\tilde{\beta}_{m-1}^{b(m)}(n-1) + s_{m-1}^{b(m)}(n)\tilde{f}_{m-1}^{(m)T}(n) \tag{62}$$

$$\tilde{b}_m^{(m)T}(n) = c_{m-1}^b(n)\tilde{f}_{m-1}^{(m)T}(n)-s_{m-1}^b(n)\lambda^{1/2}\tilde{\beta}_{m-1}^{b(m)}(n-1) \tag{63}$$

$$\tilde{q}_m(n)=s_{m-1}^b(n)\tilde{\alpha}_{m-1}(n-1) \tag{64}$$

$$\tilde{\alpha}_m(n)=c_{m-1}^b(n)\tilde{\alpha}_{m-1}(n-1) \tag{65}$$

이상으로 블록 형태의 QR 분해 순환 최소 자승 쌍선형 격자 필터 알고리듬을 얻을 수 있다. 표1.에 블록 형태의 QR 분해 순환 최소 자승 쌍선형 격자 필터 알고리듬을 나타내었다. 또한 표 1.의 블록 형태의 알고리듬은 최소 자승 계열의 알고리듬이지만 역행렬의 계산은 필요하지 않다.

식 (36)에서 $\tilde{b}_{m-1}(n-1)$의 첫 번째 요소 $\tilde{b}_{m-1,[1]}(n-1)$을 제거하기 위한 직교화 과정은 다음과 같다.

$$\begin{bmatrix} r_{m-1}^{b/2}(1,1) & r_{m-1}^{b/2}(1,2) & \cdots & r_{m-1}^{b/2}(1,L_m) \\ 0 & r_{m-1}^{b/2}(2,2) & \cdots & r_{m-1}^{b/2}(2,L_m) \\ \vdots & & \ddots & \vdots \\ 0 & 0 & & r_{m-1}^{b/2}(L_m,L_m) \\ 0 & b_{m-1,[2]}(n-1) & \cdots & b_{m-1,L_m}(n-1) \end{bmatrix} = \begin{bmatrix} c_{m-1}^f(1) & 0 & \cdots & s_{m-1}^f(1) \\ 0 & 1 & \cdots & 0 \\ \vdots & 0 & \ddots & \vdots \\ -s_{m-1}^f(1) & 0 & \cdots & c_{m-1}^f(1) \end{bmatrix} \begin{bmatrix} \lambda^{1/2}r_{m-1}^{b/2}(1,1) & \lambda^{1/2}r_{m-1}^{b/2}(1,2) & \cdots & \lambda^{1/2}r_{m-1}^{b/2}(1,L_m) \\ 0 & \lambda^{1/2}r_{m-1}^{b/2}(2,2) & \cdots & \lambda^{1/2}r_{m-1}^{b/2}(2,L_m) \\ \vdots & & \ddots & \vdots \\ 0 & 0 & & \lambda^{1/2}r_{m-1}^{b/2}(L_m,L_m) \\ b_{m-1,[1]}(n-1) & b_{m-1,[2]}(n-1) & \cdots & b_{m-1,L_m}(n-1) \end{bmatrix} \tag{66}$$

여기서 $r_{m-1}^{b/2}(i,j)$는 $\tilde{R}_{m-1}^{b/2}(n-2)$의 i,j번째 요소를 나타내고, $r_{m-1}^{b/2}(i,j)$는 $\tilde{R}_{m-1}^{b/2}(n-1)$의 i,j번째 요소를 나타낸다. 그리고 $s_{m-1}^f(i)$, $c_{m-1}^f(i)$는 시간 갱신된 $\tilde{Q}_m^f(n-1)$의 요소로 첫번째 $\tilde{b}_{m-1}(n-1)$의 요소를 제거할 때 사용한다.

위의 식에서 $\tilde{b}_{m-1,[1]}(n-1)$을 제거하기 위한 cosine 및 sine 파라미터를 구하면 다음과 같이 된다.

$$r_{m-1}^b(1,1)=\lambda r_{m-1}^b(1,1)+[\ \tilde{b}_{m-1,[1]}(n-1)]^2 \tag{67}$$

$$c_m^f(1)=\frac{\lambda^{1/2}r_{m-1}^{b/2}(1,1)}{\sqrt{r_{m-1}^b(1,1)}} \tag{68}$$

$$s_m^f(1)=\frac{\tilde{b}_{m-1,[1]}(n-1)}{\sqrt{r_{m-1}^b(1,1)}} \tag{69}$$

표 1.의 쌍선형 격자 필터의 QRD 알고리듬은 스칼라 처리를 할 수 있는 장점이 있고 알고리듬의 형태를 스칼라 처리하는 형태로 나타낼 수 있으나 매우 많은 지면을 차지하기 때문에 블록형태의 알고리듬으로 나타내었다.

표 1. 쌍선형 격자 필터의 QRD 알고리듬
Table 1. QRD algorithms for Bilinear lattice filters

Initialization{all variable:=0}

DO(T1-1) for $m=1,\cdots,N$

$c^f_m(0)=1.0,\ s^f_m(0)=0$ (T-1)

DO (T1-2) - (T1-32), for $n=1,2,\cdots$

$\tilde{f}_0(n)=x(n),\ \tilde{b}_0(n)=x(n),\ \tilde{\alpha}_0(n)=1.0,\ \tilde{e}_0(n)=y(n)$ (T-2)

$\tilde{f}^{(p)}_0(n)=\begin{cases}[\ d(n-1),x(n)d(n-1),x(n-1)d(n-1)\ ]^T\ \dagger, p=1\\ [\ x(n-p)d(n-1),x(n)d(n-p)\ ]^T\ \dagger, p=2,3,\cdots,N-1\end{cases}$ (T-3)

Iteration Procedure

DO (T-1.4) - (T-1.20), for $m=1,2,\cdots,N-1$

*Forward sine and cosine parameter

$$\tilde{R}^b_{m-1}(n)=\lambda\tilde{R}^b_{m-1}(n-1)+[\ b^T_{m-1}(n)]^2 \quad \text{(T-4)}$$

$$c^f_{m-1}(n)=\lambda^{1/2}\tilde{R}^{b/2}_{m-1}(n-1)\,\tilde{R}^{-b/2}_{m-1}(n) \quad \text{(T-5)}$$

$$s^f_{m-1}(n)=\tilde{b}^T_{m-1}(n)\,\tilde{R}^{-b/2}_{m-1}(n) \quad \text{(T-6)}$$

*Backward sine and cosine parameter

$$\tilde{R}^f_{m-1}(n)=\lambda\tilde{R}^f_{m-1}(n-1)+[\ f^T_{m-1}(n)]^2 \quad \text{(T-7)}$$

$$c^b_{m-1}(n)=\lambda^{1/2}\tilde{R}^{f/2}_{m-1}(n-1)\,\tilde{R}^{-f/2}_{m-1}(n) \quad \text{(T-8)}$$

$$s^b_{m-1}(n)=\tilde{f}_{m-1}(n)\,\tilde{R}^{-f/2}_{m-1}(n) \quad \text{(T-9)}$$

*Forward prediction error

$$\tilde{\beta}^f_{m-1}(n)=c^f_{m-1}(n-1)\lambda^{1/2}\tilde{\beta}^f_{m-1}(n-1)+s^f_{m-1}(n-1)\,\tilde{f}^T_{m-1}(n) \quad \text{(T-10)}$$

$$\tilde{f}^T_m(n)=-s^f_{m-1}(n-1)\lambda^{1/2}\tilde{\beta}^f_{m-1}(n-1)+c^f_{m-1}(n-1)\,\tilde{f}^T_{m-1}(n) \quad \text{(T-11)}$$

*Backward prediction error

$$\tilde{\beta}^b_{m-1}(n)=c^b_{m-1}(n)\lambda^{1/2}\tilde{\beta}^b_{m-1}(n-1)+s^b_{m-1}(n)\,\tilde{b}^T_{m-1}(n-1) \quad \text{(T-12)}$$

$$\tilde{b}^T_m(n)=-s^b_{m-1}(n-1)\lambda^{1/2}\tilde{\beta}^b_{m-1}(n-1)+c^b_{m-1}(n-1)\,\tilde{b}^T_{m-1}(n-1) \quad \text{(T-13)}$$

*Forward aux. prediction error

DO (T-1.14) - (T-1.15), for $p=m+1,m+2,\cdots,N-1$

$$\tilde{\beta}^{f(m)}_{m-1}(n)=c^f_{m-1}(n-1)\lambda^{1/2}\tilde{\beta}^{f(m)}_{m-1}(n-1)+s^f_{m-1}(n-1)\,\tilde{f}^{(m)T}_{m-1}(n) \quad \text{(T-14)}$$

$$\tilde{f}^{(m)T}_m(n)=-s^f_{m-1}(n-1)\lambda^{1/2}\tilde{\beta}^{f(m)}_{m-1}(n-1)+c^f_{m-1}(n-1)\,\tilde{f}^{(m)T}_{m-1}(n) \quad \text{(T-15)}$$

*Backward aux. prediction error

$$\tilde{\beta}^{b(m)}_{m-1}(n)=c^b_{m-1}(n)\lambda^{1/2}\tilde{\beta}^{b(m)}_{m-1}(n-1)+s^b_{m-1}(n)\,\tilde{f}^{(m)T}_{m-1}(n) \quad \text{(T-16)}$$

$$\tilde{b}^{(m)T}_m(n)=-s^b_{m-1}(n)\lambda^{1/2}\tilde{\beta}^{b(m)}_{m-1}(n-1)+c^b_{m-1}(n)\,\tilde{f}^{(m)T}_{m-1}(n) \quad \text{(T-17)}$$

*Joint-process estimation error

$$\tilde{\beta}_{m-1}(n)=c^f_{m-1}(n)\lambda^{1/2}\beta_{m-1}(n-1)+s^f_{m-1}(n)\,\tilde{e}_{m-1}(n) \quad \text{(T-18)}$$

$$\tilde{e}_m(n)=-s^f_{m-1}(n)\lambda^{1/2}\beta_{m-1}(n-1)+c^f_{m-1}(n)\,\tilde{e}_{m-1}(n) \quad \text{(T-19)}$$

*Angle variable

$$\tilde{\alpha}_m(n)=c^f_{m-1}(n)\,\tilde{\alpha}_{m-1}(n) \quad \text{(T-20)}$$

*Calculate $\bar{f}_N(n),\ \bar{b}_N(n)$

$$\bar{f}_N(n)=\begin{bmatrix}\tilde{f}_m(n)\\ \tilde{f}^{(m)}_m(n)\end{bmatrix} \quad \text{(T-21)}$$

$$\bar{b}_N(n)=\begin{bmatrix}\tilde{b}_m(n)\\ \tilde{b}^{(m)}_m(n)\end{bmatrix} \quad \text{(T-22)}$$

*Forward sine and cosine parameter

$$\tilde{R}^b_{N-1}(n)=\lambda\tilde{R}^b_{N-1}(n-1)+[\ \bar{b}^T_{N-1}(n)]^2 \quad \text{(T-23)}$$

$$c^f_{N-1}(n)=\lambda^{1/2}\tilde{R}^{b/2}_{N-1}(n-1)\,\tilde{R}^{-b/2}_{N-1}(n) \quad \text{(T-24)}$$

$$s^f_{N-1}(n)=\bar{b}^T_{N-1}(n)\,\tilde{R}^{-b/2}_{N-1}(n) \quad \text{(T-25)}$$

*Joint-process estimation error

$$\tilde{\beta}_{N-1}(n)=c^f_{N-1}(n)\lambda^{1/2}\tilde{\beta}_{N-1}(n-1)+s^f_{N-1}(n)\,\tilde{e}_{N-1}(n) \quad \text{(T-26)}$$

$$\tilde{e}_N(n)=-s^f_{N-1}(n)\lambda^{1/2}\beta_{N-1}(n-1)+c^f_{N-1}(n)\,\tilde{e}_{N-1}(n) \quad \text{(T-27)}$$

*Angle variable

$$\tilde{\alpha}_N(n)=c^f_{N-1}(n)\,\tilde{\alpha}_{N-1}(n) \quad \text{(T-29)}$$

*Posteriori estimation error

$$e_N(n)=\tilde{\alpha}_N(n)\,\tilde{e}_N(n). \quad \text{(T-30)}$$

† For output error formulation, $d(n)=y(n)-e(n)$

† For equation error formulation, $d(n)=y(n)$

Ⅳ. 컴퓨터 시뮬레이션

본 논문에서 제안한 알고리듬의 성능을 조사하기 위하여 식오차 방법과 출력오차 방법에 대해 미지의 쌍선형 시스템에 대한 시스템 식별 실험을 수행하였다.

적응 필터의 입력 신호 $x(n)$은 전달 함수가 식 (70)과 같은 저역 통과 필터에 평균이 0인 백색 가우시안 잡음을 통과시켜 얻었고 출력 신호의 분산은 1이 되도록 조정하였다.

$$H(z)=\frac{1}{1-1.6z^{-1}+0.95z^{-2}} \quad (70)$$

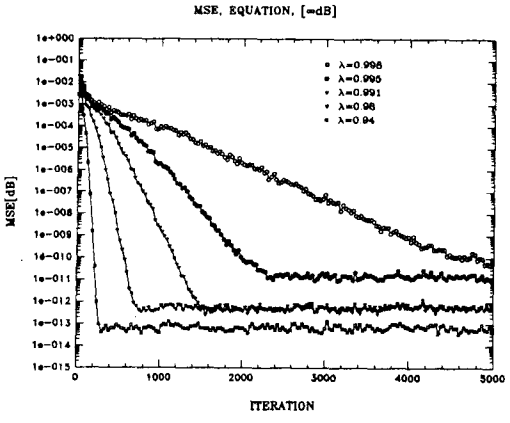
원하는 응답 신호 $d(n)$은 표 2.에 있는 계수를 이용하여 구한 다음 S/N비가 ∞dB, 30dB, 20dB가 되도록 측정 잡음을 첨가하여 얻었다. 이 측정 잡음은 평균이 0인 백색 가우시안 잡음으로 입력 신호와는 상관관계가 없는 것으로 선택하였다. 각 경우에 50번의 독립된 실험을 수행하여 앙상블 평균(ensemble average)을 구하였다.

표 2. 시스템 식별에 사용한 미지의 쌍선형 계수
Table 2. Coefficients of the unknown bilinear filter used in the system identification.

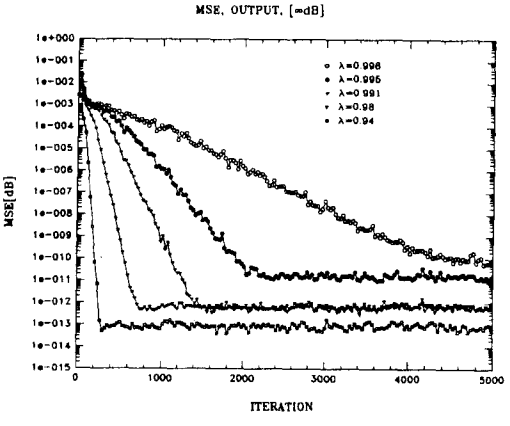
$a_0=1.0$	$a_1=1.0$	$a_2=1.0$
$c_{01}=0.3$	$c_{11}=-0.2$	$c_{21}=0.1$
$c_{02}=0.1$	$c_{12}=-0.2$	$c_{22}=0.3$
	$b_1=0.5$	$b_2=-0.5$

그림 1은 S/N비가 ∞일 때 식오차 방법과 출력오차 방법에 대한 대한 학습 곡선을 나타낸다. 가중치 요소 λ가 클 때는 학습 곡선의 수렴이 느리고 평균 자승 오차의 값도 크게 나타남을 알수 있고, λ이 작을 때는 학습 곡선의 수렴이 빠르고 평균 자승 오차의 값도 작게 나타남을 알 수 있다. 이것은 가중치 요소가 알고리듬의 시정수에 영향을 주기 때문에 발생하는 효과로

생각된다. 관측잡음이 있는 경우들의 결과들은 참고문헌[18]에서 볼수 있다.



(a)



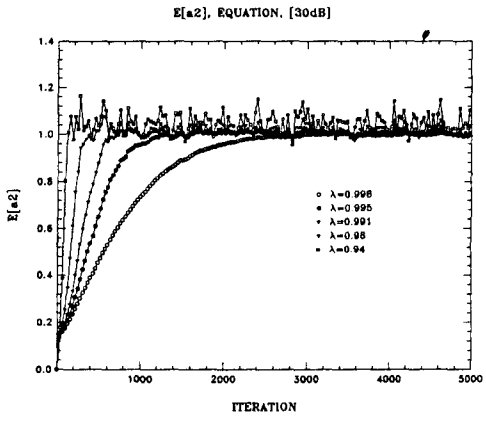
(b)

그림 1. 학습곡선
(a) 식오차 알고리듬 (b) 출력오차 알고리듬
Fig. 1. Learning curves
(a) Equation error algorithm
(b) Output error algorithms

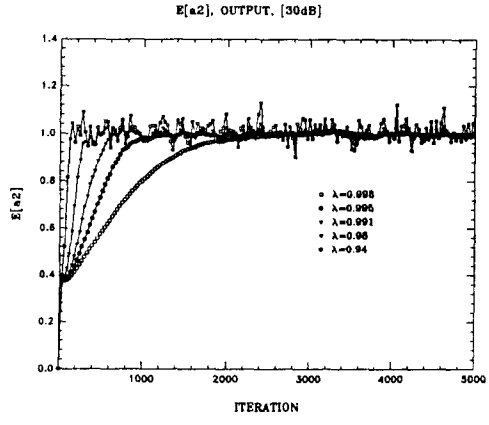
그림 2 - 그림 4에서는 S/N 비가 30dB인 경우 각 가중치 요소에 대한 각 계수의 궤적을 나타내었다.

그림 2에는 $E[a_2]$ 를 나타내었고, 그림 3에는 $E[b_1]$ 을 그리고 그림 4에는 $E[c_{01}]$ 를 나타내었다.

그림 2에서 $E[a_2]$ 의 곡선은 식오차 방법과 출력오차 방법에서 1.0 정도에 수렴함을 알 수 있고 λ가 클 때는 수렴을 느리고 하고, 곡선의 변화는 심하지 않고, λ가 작은 경우에는 수렴을 빨리하고 곡선의 변화는 심함을 알 수 있다.



(a)



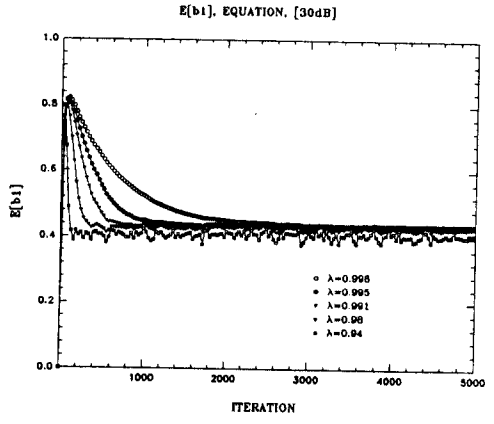
(b)

그림 2. 계수 $a_2(n)$의 평균 궤적
(a) 식오차 알고리듬 (b) 출력오차 알고리듬
Fig. 2. Mean trajectories of coefficient $a_2(n)$
(a) Equation error algorithm
(b) Output error algorithm

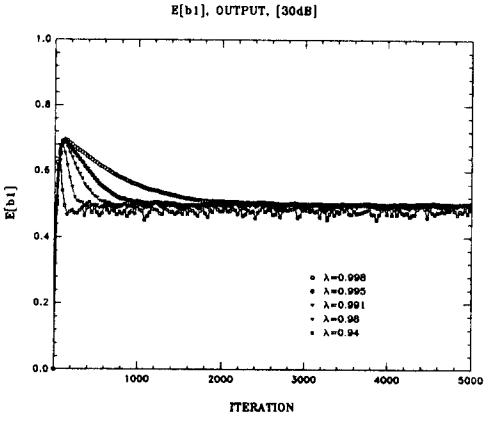
그림 3에서 $E[b_1]$ 의 곡선이 식 오차 방법에서는 0.4에 수렴하나 출력 오차 방법에서는 0.5에 수렴함을 알 수 있다. 또한 λ가 클때는 수렴을 느리고 하고, 곡선의 변화는 심하지 않고, λ가 작은 경우에는 수렴을 빨리하고 곡선의 변화는 심함을 알 수 있다. 이것은 식오차 알고리듬이 관측잡음이 있는 경우 바이어스되기 때문이다.

그림 4에서 $E[c_{01}]$ 는 식오차 방법과 출력오차 방법에서 -0.2 정도에 수렴함을 알 수 있고 λ가 클때는 수렴을 느리고 하고, 곡선의 변화는 심하지 않고, λ가

작은 경우에는 수렴을 빨리하고 곡선의 변화는 심함을 알 수 있다.



(a)

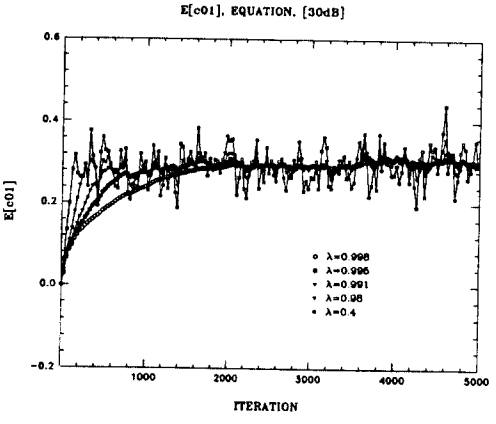


(b)

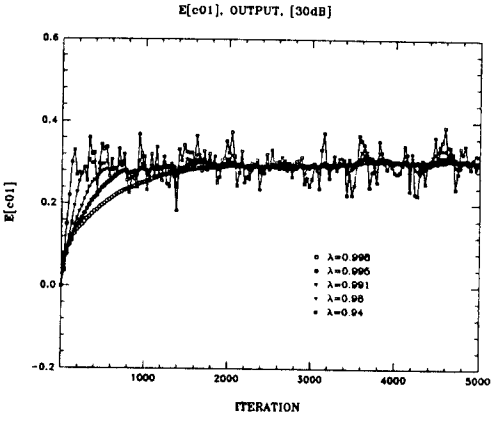
그림 3. 계수 $b_1(n)$의 평균 궤적
(a) 식오차 알고리듬 (b) 출력오차 알고리듬
Fig. 3. Mean trajectories of coefficient $b_1(n)$
(a) Equation error algorithm
(b) Output error algorithm

이상의 결과로 식오차 알고리듬은 단 하나의 최소점으로 수렴하는 장점이 있으나 관측 잡음이 있는 경우 미지 시스템의 추정이 바이어스(bias)되는 단점이 있고 그리고 출력오차 알고리듬은 현재의 출력을 얻기 위하여 과거의 적응 필터의 출력을 이용하므로 준 최적의 최소 자승의 해를 제공 받을 수 있음을 알 수 있다.

또한 misadjustment는 식오차 및 출력오차 알고리듬 가중치 요소 λ가 작을수록 커짐을 알 수 있다. 그러므로 가중치 요소 λ을 작게 하면 시정수는 작아져 수렴을 빨리 하는 반면에 misadjustment는 상대적으로 커지는 상호 보완 관계가 성립함을 알 수 있다[21].



(a)



(b)

그림 4. 계수 $c_{01}(n)$의 평균 궤적
(a) 식오차 알고리듬 (b) 출력오차 알고리듬
Fig. 4. Mean trajectories of coefficient $c_{01}(n)$
(a) Equation error algorithm
(b) Output error algorithm

V. 결 론

QR 분해법을 이용한 쌍선형 필터의 격자 알고리듬이 본 논문에서 제시되었다.

제시된 알고리듬은 식오차 알고리듬과 출력오차 알고리듬으로 유도되었으며, Givens 회전을 계산할 때 square-root가 필요로 한다.

본 논문에서 제시한 채널 분해 방법은 Non-FIR 필

터나 비선형 필터의 격자식을 쉽게 유도할 수 있을 뿐만 아니라 다양한 알고리듬을 적용시킬 수 있는 장점을 지니고 있다.

그리고 유한 어장 효과(finite wordlength effect)에 강인한 회전 연산자를 이용하여 데이터 행렬을 삼각 행렬 화시키므로 계산의 효율성이 높고 수치적 안정도가 좋다.

또한 PE를 이용하여 병렬 처리에 적합한 시스톨릭 어레이로 직접형 필터나 격자 필터를 구현할 수 있다.

식오차 알고리듬과 출력오차 알고리듬에서 관측 잡음이 있는 경우 출력오차 알고리듬이 식오차 알고리듬에 비하여 강인함을 알 수 있었고, 가중치 요소인 λ의 크기에 따라 알고리듬의 수렴 속도가 향상되나 이것에 반하여 misadjustment가 커짐도 4장의 제시된 결과로부터 알 수 있었다.

또한 각 시뮬레이션 결과를 비교하여 보면 우수한 수렴 특성과 수치적 안정도를 나타냄을 알 수 있다.

앞으로 square-root-free 알고리듬과 다양한 형태의 응용에 적용하여 알고리듬의 성능을 평가하고, 식오차 알고리듬과 출력오차 알고리듬에 대한 수치적 안정도와 수렴 특성의 이론적 해석이 이론적으로 수행되어야 할 것으로 생각된다.

참 고 문 헌

[1] J. R. Casar-Corredera, M. Carcia-Otera, and A. Figeiras-Videl, "Data echo nonlinear cancellation," *Proc. Int. Conf. Acoust., Speech, Signal Processing (Tampa, Florida)*, pp. 32.4.1-4, Mar., 1985.

[2] G. L. Sicouranza, A. Bucconi, and P. Mitri, "Adaptive echo cancellation with nonlinear digital filters," *Proc. Int. Conf. Acoust., Speech, Signal Processing (San Francisco, California)*, pp. 3.10.1-4, Mar., 1984.

[3] M. J. Coker and D. M. Simkins, "A nonlinear adaptive noise canceller," *Proc. Int. Conf Acoust., Speech, Signal Processing*, pp.470-473, 1980.

[4] R. W. Brockett, "Volterra series and geometric control theory," *Automatica*, vol. 12, pp. 167-176, 1976.

[5] C. S. Kubrusly and O. L. V. Costa, "Mean square stability conditions for discrete stochastic bilinear systems," *IEEE Trans. Automat. Contr.*, vol. AC-30, no. 11, pp.1082-1087, Nov., 1985.

[6] J. Liu, "On the existence of a general multiple bilinear time series," *J. Time Series Anal.*, vol. 10, no. 4, pp.341-355, 1989.

[7] 백홍기, 황지원, 안봉만, "적응 쌍선형 RPEM 알고리듬," 대한전자공학회 논문지, 제30권, B편, 제3호, pp.10-21, 3월, 1993

[8] J. Lee and V. J. Mathews, "Output-error LMS bilinear filters with stability monitoring," *Proc. Int. Conf. Acoust., Speech Signal processing(Detroit, Michigan)*, vol. 2, pp. 965-968, 1995.

[9] S. Haykin, *Adaptive Filter Theory*, Englewood Cliffs, N. J., Prentice Hall, 1991.

[10] J. G. Proakis, C. M. Rader, F. Ling and C. L. Nikias, *Advanced Digital Signal Processing*, Macmillan Publishing Company.

[11] N. Kalouptsidis and S. Theodoridis, *Adaptive System Identification and Signal Processing Algorithms*, Prentice Hall International (UK) Limited, 1993.

[12] W. M. Gentleman and H. T. Kung, "Matrix triangularization by systolic arrays," *Proc. SPIE, Real Time Signal Processing 6*, vol. 298, pp. 298, 1981.

[13] J. G. McWhirter, "Recursive least-squares minimization using a systolic array," *Proc. SPIE., Real Time Signal Processing 6*, vol.431, pp.105-112, 1983.

[14] F. Ling and J. G. Proakis, "A generalized multichannel least squares lattice algorithm based on sequential pro-

cessing stages," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, no. 2, pp.381-389, 1984.

[15] M. A. Syed and V. J. Mathews, "QR-Decomposition based algorithms for adaptive Volterra filters," *IEEE Trans. on Circuits and Systems I. Fundamental Theory and Applications*, vol. 40, no. 6, pp. 372-382, June 1993.

[16] M. A. Syed and V. J. Mathews, "Lattice algorithms for recursive least squares adaptive Volterra filtering," *IEEE Trans. on Circuits and Systems II. Analog and Digital Signal Processing*, vol. 41, no. 3, pp. 202-214, March, 1994.

[17] H. K. Baik and V. J. Mathews, "Adaptive lattice bilinear filters," *IEEE Trans. on Signal Processing*, vol. 41, no. 6, pp.2033-2046, June, 1993.

[18] 안봉만, 황지원, 백홍기, "QR 분해법을 이용한 적응 쌍선형 격자 알고리듬," 대한전자공학회 논문지, 제31권, B편, 제10호, pp.32-43, 10월, 1994.

[19] V. J. Mathews and J. Lee, "A fast least-squares second-order Volterra filter," *Proc. Int. Conf. Acoust., Speech, Signal Processing (New York)*, 1988.

[20] B. Yang and J. F. Ohme, "Rotation based RLS algorithms: Unified derivation, numerical properties and parallel implementations," *to appear in the IEEE Trans. on Signal Processing.*

[21] E. Eleftheriou and D. D. Falconer, "Tracking properties and steady state performance of RLS adaptive filter algorithms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, no. 5, pp.1097-1110, Oct., 1986.

저 자 소 개

安 奉 萬(正會員)

1962年 9月 17日生. 1988年 2月 全北大學校 工科大學 電子工學科 졸업(공학사). 1990年 2月 全北大學校 大學院 電子工學科 졸업(공학석사). 1990年 5月 ~ 1992年 2月 현대정공 기술연구소 연구원. 1994年 2月 全北大學校 大學院 電子工學科 박사과정 수료. 1995年 3月 ~ 현재 井邑工業專門大學 電子通信科 전임강사. 주관심분야는 신호처리, 적응신호처리, 음성신호처리, 비선형 신호처리 분야 등임.

白 興 基(正會員)

1955年 1月 5日生. 1977年 2月 서울大學校 工科大學 電子工學科 졸업, 공학학사 취득. 1979年 8月 서울大學校 大學院 電子工學科 졸업, 공학석사 취득. 1987年 2月 서울大學校 大學院 電子工學科 졸업, 공학박사 취득. 1981年 3月 ~ 현재 全北大學校 工科大學 電子工學科 교수. 1990年 1月 ~ 1990年 12月 미국 Utah 대학 Post Doc. 주관심분야는 디지탈 신호처리, 적응신호처리, 비선형 신호처리 분야등임.