적응 파라미터 예측을 위한 근사화된 RLS 알고리즘

정회원 안 봉 만*, 황 지 원**, 유 정 래***, 조 주 필****

An Approximated RLS Algorithm for Adaptive Parameter Estimation

Bong-Man Ahn*, JeeWon Hwang**, Jung Rae Ryoo*** Juphil Cho**** ***Regular Members***

요 약

본 논문은 근사화 기법을 RLS 알고리즘에 적용한 고속 적응 알고리즘을 제안한다. 제안 알고리즘(D-RLS)은 QR 분해 RLS 알고리즘 유도 과정을 RLS 알고리즘으로부터 역으로 유도한 알고리즘이다. 유도된 알고리즘(D-RLS)은 입력 신호들이 서로 분리되어 있다는 가정을 사용한 알고리즘과 유사한 형태를 취한다. 이 알고리즘의 계산량은 $O(N^2)$ 보다 작은 $O(N)$이다. 이 알고리즘의 성능 평가를 위하여 FIR 시스템과 비선형(Volterra) 시스템의 시스템 식별 기법을 이용하였으며, 결과적으로 우수한 성능을 나타냄을 확인하였다.

Key Words : Fast adaptive algorithm, System identification, Nonlinear system(Volterra) identification.

ABSTRACT

This paper presents the fast adaptive algorithm which applies an approximation scheme into RLS algorithm. The proposed algorithm(D-RLS) derives a QRD RLS algorithm derivation process from RLS algorithm recursively. D-RLS has the similar pattern as the algorithm having the approximation that input signals are separated respectively. Computational complexity of D-RLS is $O(N)$, fewer than $O(N^2)$. To evaluate performance of proposed algorithm, we use the system identification method of FIR and Volterra system. And, finally, we can show D-RLS has an excellent performance.

Ⅰ. 서 론

많은 적응 필터 응용 분야에서 계산량을 줄이려는 노력들이 진행되고 있다. 이중 계산량이 적은 알고리즘은 LMS 계열의 알고리즘들이다. 그러나 LMS 계열 알고리즘들의 성능은 입력 신호들의 상관관계에 따라서 수렴속도가 결정되는 단점이 있다. LMS 계열 알고리즘에서 이러한 문제점들을 보완하고자 많은 시도들이 있었다. 그중 LMS 알고리즘의 적응상수(step-size)를 시간에 따라 변화시켜 알고리즘의 수렴속도를 향상시킨 논문들도 있다[1-4]. 이러한 논문들 중에 알고리즘의 초기변수에 민감하거나 행렬 연산이 필요한 논문들도 있다. LMS 계열 알고리즘에서 시변 적응상수를 적용할 때 또 다른 파리미터를 정의하야 하는 단점이 있다.

반면에 LS 계열 알고리즘은 우수한 수렴속도는 유지하면서 계산량을 줄이는 방법에 대한 논문들이 많이 발표되었다[5-6]. 계산량이 적은 Fast RLS 알고리즘은 수치적으로 안정하지 못한 단점을 지니고 있어 이것을 해결하는 하나의 방법으로 QR 분해가

※ 본 연구는 한국과학재단 특정기초연구(No. R01-2006-000-11183-0(2007)) 지원으로 수행되었음.
* 전북대학교 Next 사업단, **익산대학 컴퓨터과학과, ***서울산업대학교 제어계측공학과,
**** 교신저자, 군산대학교 전자정보공학부(stefano@kunsan.ac.kr),
논문번호 : KICS2007-02-057, 접수일자 : 2007년 2월 12일, 최종논문접수일자 : 2007년 7월 15일

대두되기 시작하였다[7-9]. 그러나 좋은 수치적 안정도를 보이는 일반적인 QR 분해 방법 역시 계산량은 $O(N^2)$이다. 그리고 많은 논문들에서 QR 분해 방법을 이용한 고속 알고리즘의 개발이 보고되어 왔다. QR 분해에 의한 RLS 구현이 주로 오차를 계산하기 때문에 계수벡터를 산출하기 위해서 다른 수학적 도구들이 필요하다. 이 계산하는 데 필요한 계산량은 $O(N^2)$이다[10-12].

참고문헌 [11]에서 QR 분해에서 입력 신호들이 Givens 회전 행렬에 의하여 상관관계가 없다(decorreate)는 가정에서 계산량이 $O(N)$인 QR 계열의 RLS 알고리즘(A-QR-RLS)을 개발하였다. 참고문헌 [11]은 입력신호 들이 상관관계가 없다는 가정을 직접적으로 사용하지 않고 알고리즘의 유도하는 중에 사용하여 그 사용처가 직접적으로 나타나지는 않는다. 본 논문 역시 입력 신호들이 서로 분리되어 있다는 가정을 사용하였다. 입력신호들이 Givens 회전 행렬에 의하여 회전되어 상관관계가 없다고 가정하였다. 이것은 참고문헌 [11]과 같은 가정이다. 이와 같은 가정은 데이터 행렬을 QR 분해한 결과 행렬 $R(n-1)$을 $n-1$에서 대각행렬로 구성할 수 있게 된다. 그러나 반복시간 n에서 $R(n)$은 직교 행렬 Q가 곱해지기 때문에 $R(n)$은 더 이상 대각행렬이 아닌 행렬이 된다. 그래서 입력신호들이 서로 상관관계가 없다는 가정과 반복시간 $n-1$ 대각행렬 된다는 사실을 유지하면서 반복시간 n에서도 그 결과가 대각행렬인 방법을 구상하였다. 이와 같은 방법으로 알고리즘을 유도하면 참고문헌[10,11]의 알고리즘보다 계산량이 적게 된다. 그 방법 중 하나로 QR 분해 방법을 이용한 순환 최소자승 알고리즘은 순환 최소 자승 알고리즘으로 변형할 수 있는데 이와 같이 하면 서로 상관관계가 없는 입력신호들과 반복시간 $n-1$에서 대각행렬의 행렬 덧셈의 결과는 반복시간 n에서 반드시 대각행렬이 된다는 사실을 이용하여 본 논문에서 제안한 알고리즘을 유도하였다. 제안한 알고리즘의 성능 평가를 위해 시변 계수를 갖는 비선형 Volterra 시스템의 시별 문제에 적용하여 여러 알고리즘과 그 성능을 비교하였다. 평가에 참여한 알고리즘은 NLMS (Normalized Least Mean Square), RLS 및 LMS 알고리즘이다. 그 결과 본 논문에서 제안한 알고리즘은 RLS 보다는 성능이 떨어지지만 NLMS와 유사하였다. 본 논문의 구성은 Ⅱ절에서 알고리즘의 유도에 관련한 사항들을 다루었고, Ⅲ절에서는 컴퓨터 시뮬레이션을 통하여 알고리즘의 성능을 확인하였으며 Ⅳ절에서 결론 및 논의를 다루었다.

Ⅱ. 본 론

2.1 RLS 알고리즘의 고찰

RLS 알고리즘은 다음과 같은 비용함수를 최소화하여 얻어진다.

$$\xi_i(n)=\sum_{i=1}^{n}\lambda^{(n-i)/2}\|e(i)\|^2 \tag{1}$$

여기서 $e(n)$은 오차로 다음과 같다.

$$e(n)=d(n)-\ w_o^H u_M(n) \tag{2}$$

비용함수를 최소화하는 계수벡터는 다음과 같은 정규방정식을 최소화함으로서 얻어진다.

$$\Phi(n)\ w_R(n)=\Theta(n) \tag{3}$$

$$\Phi(n)=\lambda\,\Phi(n-1)+\ x(n)\ x^H(n) \tag{4}$$

$$\Theta(n)=\lambda\,\Theta(n-1)+\ x(n)d(n) \tag{5}$$

식(3)의 정규방정식 계산에 역행렬은 matrix inversion lemma를 이용하여 구한다. 계수벡터를 구하면 다음과 같이 된다.

$$\boldsymbol{w}_R(n)=\boldsymbol{w}_R(n-1)+\boldsymbol{k}(n)(d(n)-\boldsymbol{u}^H(n)\boldsymbol{w}_R(n-1)) \tag{6}$$

$$k(n)=\frac{\lambda^{-1}\Phi^{-1}(n-1)\ u(n)}{1+\lambda^{-1}u^H(n)\ \Phi^{-1}(n-1)\ u(n)} \tag{7}$$

일반적인 RLS 알고리즘의 계산량은 $O(N^2)$이다.

2.2 QR 분해와 RLS 알고리즘

식(3)의 정규 방정식에서 상관행렬 $\Phi(n)$은 다음과 같이 QR 분해 될 수 있다.

$$\Phi(n)=\ R^H(n)\ Q^H(n)\ Q(n)R(n) \tag{8}$$

여기서 $R(n)$은 상삼각 행렬(upper triangular matrix)이고, 상관행렬 $\Phi(n)$을 Cholesky 분해하였을 때 유일한 Cholesky 요소이다. 그리고 $Q(n)$은 직교행렬이다. 식(8)를 이용하여 식(3)의 정규 방정

식을 나타내면 다음과 같다.

$$\boldsymbol{R}^H(n)\boldsymbol{R}(n)\hat{\boldsymbol{w}}_R(n)=\boldsymbol{\theta}(n) \tag{9}$$

여기서 위의 식을 간단히 표현하기 위하여 아래의 $\Gamma(n)$을 정의하자.

$$\boldsymbol{R}(n)\hat{\boldsymbol{w}}(n)=\boldsymbol{\Gamma}(n) \tag{10}$$

$$\boldsymbol{R}^H(n)\boldsymbol{\Gamma}(n)=\boldsymbol{\theta}(n) \tag{11}$$

식(10)에서 계수벡터는 다음과 같이 구할 수 있다.

$$\hat{\boldsymbol{w}}_R(n)=\boldsymbol{R}^{-1}(n)\boldsymbol{\Gamma}(n) \tag{12}$$

식(10)을 순환 과정은 다음과 같이 나타낼 수 있다.

$$\boldsymbol{Q}(n)\begin{bmatrix}\lambda^{1/2}\boldsymbol{R}(n-1) & \lambda^{1/2}\boldsymbol{\Gamma}(n-1) & \boldsymbol{0}\\ \boldsymbol{x}^H(n) & d(n) & 1\end{bmatrix}=\begin{bmatrix}\boldsymbol{R}(n) & \boldsymbol{\Gamma}(n) & \boldsymbol{\beta}(n)\\ \boldsymbol{0}^H & \tilde{e}(n) & \alpha(n)\end{bmatrix} \tag{13}$$

참고 문헌 [11]의 과정을 식(13)에 적용하여 보자. 식(13)에서 $\boldsymbol{R}(n-1)$과 $\boldsymbol{x}(n)$의 항으로 행렬을 분리하고 계수벡터 $\hat{\boldsymbol{w}}_R(n)$을 곱하는 형태로 나타낼 수 있다.

$$\boldsymbol{Q}(n)\begin{bmatrix}\lambda^{1/2}\boldsymbol{R}(n-1)\\ \boldsymbol{x}_M^H(n)\end{bmatrix}\begin{bmatrix}\hat{\boldsymbol{w}}_R(n-1)\\ \boldsymbol{0}^T\end{bmatrix}=\begin{bmatrix}\boldsymbol{R}(n)\\ \boldsymbol{O}\end{bmatrix}\begin{bmatrix}\hat{\boldsymbol{w}}_R(n)\\ \boldsymbol{0}^T\end{bmatrix} \tag{14}$$

식 (14)가 Givens 회전할 때 decorrelated 되는 경우 식은 다음과 같이 행렬 형태로 쓸 수 있다.

$$\begin{bmatrix}r_{1,1}(n-1) & 0 & \cdots & 0\\ 0 & r_{2,2}(n-1) & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & r_{M,M}(n-1)\end{bmatrix}\begin{bmatrix}w_1\\ w_2\\ \vdots\\ w_M\end{bmatrix}=\begin{bmatrix}\rho_1(n-1)\\ \rho_2(n-1)\\ \vdots\\ \rho_M(n-1)\end{bmatrix} \tag{15}$$

참고문헌 [11]에서 식(15)를 Givens 회전과정에서 decoupled 되었다라고 하였다. 식(9)의 성질을 이용하기 위하여 식(13)의 순환과정 형태로 나타내자. 식(13)의 $\boldsymbol{\Gamma}(n)$와 식(16)의 $\tilde{\boldsymbol{p}}(n)$은 다른 것으로 가정하자.

$$\boldsymbol{Q}(n)\begin{bmatrix}\lambda^{1/2}\tilde{\boldsymbol{D}}(n-1) & \lambda^{1/2}\tilde{\boldsymbol{p}}(n-1) & \boldsymbol{0}\\ \boldsymbol{x}^H(n) & d(n) & 1\end{bmatrix}=\begin{bmatrix}\tilde{\boldsymbol{D}}(n) & \tilde{\boldsymbol{p}}(n) & \boldsymbol{\beta}(n)\\ \boldsymbol{0}^H & \tilde{e}(n) & \alpha(n)\end{bmatrix} \tag{16}$$

여기서 $\tilde{\boldsymbol{D}}(n-1)$과 $\tilde{\boldsymbol{p}}(n)$은 다음과 같이 정의된다.

$$\tilde{\boldsymbol{D}}_{i,i}(n-1)=\begin{cases}r_{i,i}(n-1) & ,i=j,\ i=1,2,\cdots,M\\ 0 & ,otherwise\end{cases} \tag{17}$$

$$\tilde{\boldsymbol{p}}^H(n)=[\rho_1(n),\rho_2(n),\cdots,\rho_M(n)] \tag{18}$$

식(16)의 $\tilde{\boldsymbol{D}}(n-1)$은 대각행렬이지만 $\boldsymbol{x}^H(n)$을 Givens 회전시키는 M개의 회전 행렬을 적용한 결과 $\tilde{\boldsymbol{D}}(n)$ 행렬은 더 이상 대각행렬이 아니다. 이것은 식(15)의 장점들이 일부 사라지게 된다. 그러므로 참고문헌[10,11]에서와 같이 식(16)을 QR 분해 방법으로 알고리즘을 구하면 비 대각행렬 $\tilde{\boldsymbol{D}}(n)$ 때문에 계산량이 증가하여 비 효율적인 방법이 된다. 효율적인 방법은 $\tilde{\boldsymbol{D}}(n-1)$와 $\tilde{\boldsymbol{D}}(n)$가 같이 대각행렬이 되는 방법이다. 그 방법으로 중 하나로 식(16)을 전치하여 식(16)의 앞에서 곱하자. 그러면 다음의 수식을 얻을 수 있다.

$$\begin{bmatrix}\lambda^{1/2}\tilde{\boldsymbol{D}}^H(n-1) & \boldsymbol{x}^H(n)\\ \lambda^{1/2}\boldsymbol{p}^H(n-1) & d(n)\end{bmatrix}\boldsymbol{Q}^H(n)\boldsymbol{Q}(n)\begin{bmatrix}\lambda^{1/2}\tilde{\boldsymbol{D}}(n-1) & \lambda^{1/2}\boldsymbol{p}(n-1)\\ \boldsymbol{x}^H(n) & d(n)\end{bmatrix}$$
$$=\begin{bmatrix}\tilde{\boldsymbol{D}}^H(n) & \boldsymbol{0}\\ \boldsymbol{p}^H(n) & \tilde{e}(n)\end{bmatrix}\begin{bmatrix}\tilde{\boldsymbol{D}}(n) & \boldsymbol{p}(n)\\ \boldsymbol{0}^H & \tilde{e}(n)\end{bmatrix} \tag{19}$$

위의 식에서 $\boldsymbol{Q}^H(n)\boldsymbol{Q}(n)=I$이다. 식(19)을 정리하면 다음의 수식들을 얻을 수 있다.

$$\tilde{\boldsymbol{D}}^T(n)\tilde{\boldsymbol{D}}(n)=\lambda\tilde{\boldsymbol{D}}^H(n-1)\tilde{\boldsymbol{D}}(n-1)+\tilde{\boldsymbol{x}}^H(n)\tilde{\boldsymbol{x}}(n) \tag{20}$$

$$\tilde{\boldsymbol{D}}^H(n)\tilde{\boldsymbol{p}}(n)=\lambda\tilde{\boldsymbol{D}}^H(n-1)\tilde{\boldsymbol{p}}(n-1)+\boldsymbol{x}^H(n)d(n) \tag{21}$$

$$\tilde{\boldsymbol{x}}^T(n)=[x_1(n)\delta_1,x_2(n)\delta_2,\cdots,x_M(n)\delta_M] \tag{22}$$

여기서 $\delta_i,\ i=1,2,\cdots,M$은 Kronecker delta 이다. 식(20)과 식(21)을 간단히 표현 하고자 다음과 같이 변수를 정하였다.

$$\boldsymbol{D}(n)=\tilde{\boldsymbol{D}}^H(n)\tilde{\boldsymbol{D}}(n) \tag{23}$$

$$\tilde{\boldsymbol{\theta}}(n)=\tilde{\boldsymbol{D}}^H(n)\tilde{\boldsymbol{p}}(n) \tag{24}$$

$$\boldsymbol{D}(n)=\lambda\boldsymbol{D}(n-1)+\tilde{\boldsymbol{x}}^T(n)\tilde{\boldsymbol{x}}(n) \tag{25}$$

$$\tilde{\boldsymbol{\theta}}(n)=\lambda\tilde{\boldsymbol{\theta}}(n-1)+\tilde{\boldsymbol{x}}^T(n)d(n) \tag{26}$$

식(25)는 식(4)로 정의한 입력벡터의 상관행렬중 주

대각 요소로 구성된 행렬이고 식(26)은 식(5)의 상호 상관 벡터와 같다. 또한 $\boldsymbol{D}(n)$과 $\boldsymbol{D}(n-1)$은 같은 차원의 대각 행렬이다.

$$\boldsymbol{\theta}(n)=\lambda\boldsymbol{\theta}(n-1)+\tilde{\boldsymbol{x}}^{T}(n)d(n) \tag{27}$$

그러므로 식(25)과 식(27)는 기존 RLS 알고리즘의 제한된 표현들이다.

2.3 D-RLS 알고리즘의 유도

식(25)과 식(27)로 정의된 상관행렬과 상호 상관 벡터를 이용하여 계수벡터를 구하자. 본 논문에서는 계수벡터를 다음과 같이 구하였다.

$$\hat{\boldsymbol{w}}(n)=\boldsymbol{D}^{-1}(n)\boldsymbol{\theta}(n) \tag{28}$$

행렬 $\boldsymbol{D}(n)$가 주 대각 요소만 있는 대각 행렬이므로 직접적으로 역 행렬을 구할 수 도 있지만 본 논문에서는 $\boldsymbol{D}(n)$의 역행렬을 이용한다.

$$\boldsymbol{D}^{-1}(n)=\lambda^{-1}\boldsymbol{D}^{-1}(n-1)-\frac{\lambda^{-1}\boldsymbol{D}^{-1}(n-1)\tilde{\boldsymbol{x}}(n)\tilde{\boldsymbol{x}}^{H}(n)\boldsymbol{D}^{-1}(n-1)}{1+\lambda^{-1}\tilde{\boldsymbol{x}}^{H}(n)\boldsymbol{D}^{-1}(n-1)\tilde{\boldsymbol{x}}(n)} \tag{29}$$

$$\boldsymbol{k}(n)=\frac{\lambda^{-1}\boldsymbol{D}^{-1}(n-1)\tilde{\boldsymbol{x}}(n)}{1+\lambda^{-1}\tilde{\boldsymbol{x}}^{H}(n)\boldsymbol{D}^{-1}(n-1)\tilde{\boldsymbol{x}}(n)} \tag{30}$$

$\boldsymbol{D}(n)$의 역행렬을 $\boldsymbol{P}(n)$이라고 하면

$$\boldsymbol{P}(n)=\lambda^{-1}\boldsymbol{P}(n-1)-\lambda^{-1}\boldsymbol{k}(n)\tilde{\boldsymbol{x}}^{H}(n)\boldsymbol{P}(n-1) \tag{31}$$

벡터 $\boldsymbol{k}(n)$의 i행의 요소 $k_i(n)$를 정리하면 다음과 같다. 또한 $p_{i,i}(n)$은 행렬 $\boldsymbol{P}(n)$의 i,i번째 요소이다.

$$k_i(n)=\frac{p_{i,i}(n-1)\tilde{x}_i(n)}{1+p_{i,i}(n-1)\tilde{x}_i^{2}(n)} \tag{32}$$

위의 $k_i(n)$를 이용하여 계수벡터를 정리하면 다음과 같이된다.

$$\hat{w}_i(n)=\hat{w}_i(n-1)+k_i(n)(d(n)-\tilde{\boldsymbol{x}}(n)\hat{\boldsymbol{w}}(n-1)) \tag{33}$$

$$p_{i,i}(n)=\lambda^{-1}p_{i,i}(n-1)-\lambda^{-1}k_i(n)\tilde{x}_i(n)p_{i,i}(n-1) \tag{34}$$

이상으로 제안한 알고리즘을 정리하면 표 1과 같다. 이 알고리즘의 계산량은 $6M$으로 $O(M)$ 알고리즘이 된다. M은 필터의 차수이다.

표 1. 제안한 D-RLS 알고리즘
Table 1. Propose D-RLS algorithm

==========초기조건==========

$p_{i,i}(n-1)=\delta$, all variable =0 for n=1, 2, 3, …, N

$e(n)=d(n)-\tilde{\boldsymbol{x}}^{T}(n)\hat{\boldsymbol{w}}(n-1)$ for i=1, 2, 3, …, M

$\psi_i(n)=p_{i,i}(n-1)\tilde{x}_i(n)$

$k_i(n)=\dfrac{\psi_i(n)}{1+\psi_i(n)\tilde{x}_i(n)}$

$\hat{w}_i(n)=\hat{w}_i(n-1)+k_i(n)e(n)$

$p_{i,i}(n)=\lambda^{-1}p_{i,i}(n-1)-\lambda^{-1}k_i(n)\psi_i(n)$

end loop %i

end loop %n

==========

2.4 D-RLS 알고리즘의 수렴특성 고찰

제안된 알고리즘(D-RLS)은 기존 RLS 알고리즘의 수렴특성을 기존 해석으로 고찰할 수 있다. 기존 RLS 알고리즘은 $\lambda=1$인 경우와 $\lambda\neq1$인 경우 참고문헌[12]에서 잘 유도되어 있다.

$$E[\hat{w}_i(n)]=w_o^{(i)}+b_i(n) \tag{35}$$

$$b_i(n)=-\delta d_{ii}^{-1}(n)w_o^{(i)} \tag{36}$$

여기서 $w_o^{(i)}(n)$은 미지의 파라미터로 i번째 요소이고 $b_i(n)$은 bias이다. 또한 δ는 초기값이고 $d_{ii}(n)$은 대각행렬 $\boldsymbol{D}(n)$의 (i,i)번째 요소이다. $d_{ii}(n)$은 시간 평균을 나타내므로 앙상블 평균을 사용하면 다음의 식을 얻을 수 있다.

$$b_i(n)\approx-\frac{\delta}{n}r_{ii}^{-1}(n)w_o^{(i)},\quad \text{for } i=1,2,\cdots,M \tag{37}$$

식(37)은 반복횟수 n이 증가하면 bias는 사라짐을 의한다.

최적계수와 예측계수와의 차는 다음과 같이 구해질 수 있다.

$$E[\hat{w}_i(n)]\approx w_o^{(i)}-\frac{\sigma_o^2}{n}r_{ii}^{-1}w_0^{(i)} \tag{38}$$

여기서 σ_o^2은 측정잡음의 분산이다.

계수 벡터 오차 상관행렬은 다음과 같이 정리된다.

$$\varepsilon_i(n)=E[(w_o^{(i)}-\hat{w}_i(n))(w_o^{(i)}-\hat{w}_i(n))^{H}]\approx\frac{1}{n}\sigma_o^2\frac{1}{r_{ii}} \tag{39}$$

여기서 $\mathrm{tr}[\cdot]$은 trace를 의미하고 r_{ii}는 $\boldsymbol{R}$의 (i,i)번째 요소이다. 결과적으로 제안한 알고리즘은 식(38)과 식(39)에서 최적 계수에 수렴함을 알 수 있다.

Ⅲ. 컴퓨터 모의 실험

실험 1. FIR 시스템 식별

컴퓨터 모의실험을 위하여 다음의 FIR 시스템의 시스템 식별 문제에 적용하였다.

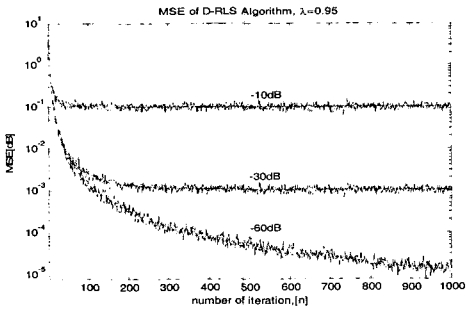
$$x(n)=0.5x(n-1)+v(n) \tag{40}$$

$$y(n)=-0.78x(n)-1.48x(n-1)+1.38x(n-2) \tag{41}$$

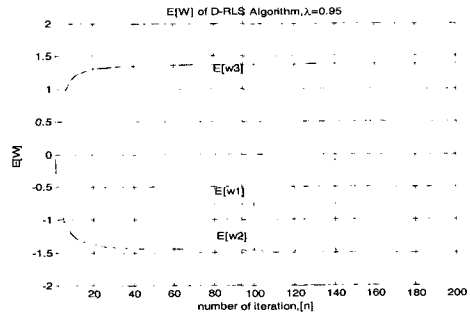
입력 $v(n)$은 평균은 0이고 분산은 1인 가우시안 잡음이다,

모든 실험은 100번의 독립된 실험을 수행한 후 앙상블 평균하여 그 결과를 나타내었다. 이때 $\lambda=0.95$, 초기값은 $\delta=1.0$으로 하였다. 그림 1은 측정잡음이 -10dB, -30dB 및 -60dB인 경우 MSE 곡선과 계수벡터의 앙상블 평균값을 각각 나타내었다. 그림 1에서 빨강색 그래프는 최종 이론식을 대신 다음의 식을 컴퓨터로 계산하여 나타 낸 것이다. 그림 1에서 MSE 곡선들은 식(42)과 같이 수렴 후에 측정 잡음의 S/N비 만큼에 수렴함을 알 수 있다.

$$J(n)=\sigma_o^2+\mathrm{tr}\left[E[\tilde{\boldsymbol{x}}(n)\tilde{\boldsymbol{x}}^H(n)]E[\boldsymbol{\varepsilon}_\delta(n-1)\boldsymbol{\varepsilon}_\delta^H(n-1)]\right] \tag{42}$$



(a) learning curves



(b)E[W] curves

그림 1. 서로 다른 S/N비를 갖는 D-RLS 알고리즘
Fig. 1. D-RLS algorithm with different S/N ratio

그림 2는 각 S/N비 별로 1000개의 데이터 중 200개의 데이터를 나타낸 것이며 최적의 값에 잘 수렴함을 알 수 있다. 그림 2는 LMS, NLMS, RLS 및 D-RLS 알고리즘의 MSE 곡선을 나타내었다. 이때 S/N비는 -30dB로 하였다. LMS 알고리즘이 속도에서 가장 느리고 RLS 알고리즘의 수렴 속도가 가장 빠름을 알 수 있다. 제안된 알고리즘은 NLMS 알고리즘 보다 수렴이 빠르고 정상상태에서 측정잡음의 분산값에 가장 근접함을 알 수 있다.

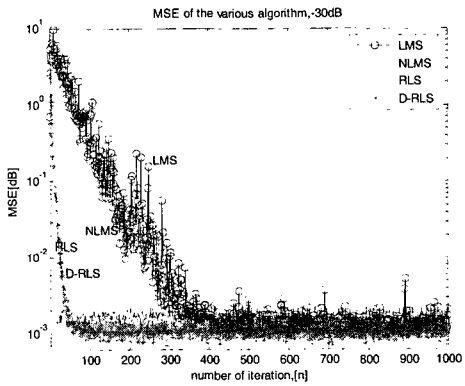


그림 2. 다양한 알고리즘의 MSE.
Fig. 2. MSE of the various algorithms

실험 2. 시변 계수를 갖는 Volterra 시스템의 식별

다음의 시뮬레이션은 비선형 시스템의 시스템 시별 문제에 적용한 결과이다. 비선형 시스템은 절단된 Volterra 시스템으로 구성하였다.

$$x(n)=v(n)+a(n)x(n-1) \tag{43}$$

$$\begin{aligned}y(n)=&-0.78x(n)+b(n)x(n-1)+1.39x(n-2)\\&+1.0x^2(n)-1.62x^2(n-1)-0.23x^2(n-2)\\&+3.72x(n)x(n-1)+0.5x(n)x(n-2)+1.86x(n-1)x(n-2)\end{aligned} \tag{44}$$

$$b(n)=-1.46+0.5\sin(2\pi nf) \tag{45}$$

입력 $v(n)$은 평균이 0이고 분산 1인 백색 잡음을 사용하였다. 그리고 측정 잡음 $v(n)$은 입력과의 S/N비를 -30dB로 하여 사용하였다. 식(44)의 입력들과 커널벡터를 다음과 같이 재 정의하였다.

$$\begin{aligned}&x(1)=x(n),\ x(2)=x(n-1),\ x(3)=x(n-2),\\&x(4)=x^2(n),x(5)=x^2(n-1),x(6)=x^2(n-2),\\&x(7)=x(n)x(n-1),x(8)=x(n-1)x(n-2),x(9)=x(n)x(n-2)\end{aligned} \tag{46}$$

$$\begin{aligned}&w_o^1=-0.78,\ w_o^2=b(n),\ w_o^3=1.39,\ w_o^4=1.0\\&w_o^5=-1.62,w_o^6=-0.23,w_o^7=3.72,\ w_o^8=1.86,w_o^9=0.5\end{aligned} \tag{47}$$

그림 3은 학습곡선을 나타낸 그림이다. 이때 식(43)의 필터 계수 $a(n)$은 0.5와 0.9를 사용하였고 $b(n)=-1.46$을 이용하였다. 각 실험은 100번의 독립된 실험을 수행해 얻은 결과들을 앙상블 평균하여 나타내었다.

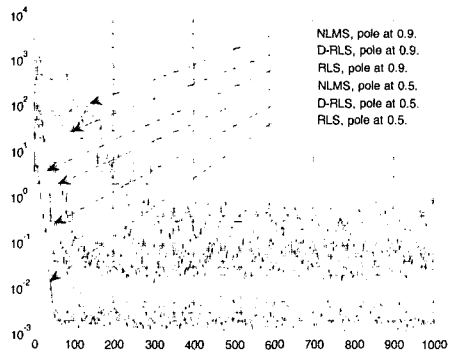


그림 3. 다양한 알고리즘들의 학습 곡선들(고정 kernel)
Fig. 3. Learning curves of various algorithms(fixed kernel)

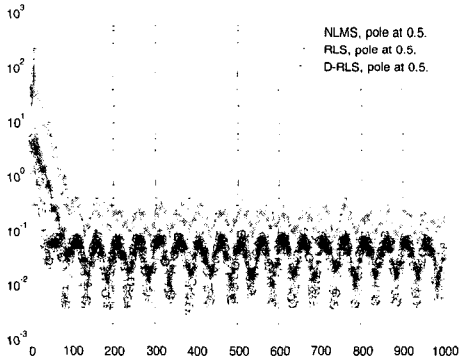


그림 4. 다양한 알고리즘의 학습곡선들.(시변 kernel)
Fig. 4. Learning curves of various algorithms(time varying kernel)

그림 3에서 RLS 알고리즘이 식(33)의 극점 위치에 관계없이 수렴속도는 가장 좋은 결과를 나타내고 있다. 그리고 D-RLS 과 NLMS 알고리즘은 극점의 위치에 따라 수렴 속도가 차이가 많이 남을 알 수 있다. 수렴 후의 학습곡선의 값은 D-RLS 알고리즘이 RLS 알고리즘에 가장 유사하거나 일부의 경우는 RLS 알고리즘 보다 더 우수함을 보이기도 한다. 비교된 알고리즘들은 colored noise를 생성하는 pole의 위치에 따라 성능에 큰 차이가 있음을 알 수 있다. 전체적인 성능에서 D-RLS알고리즘은 RLS 알고리즘 보다는 떨어지지만 NLMS알고리즘 보다는 성능이 우수함을 알 수 있다. 그림 4는 식(43)의 필터 계수 $a(n)$은 0.5, 식(34)의 $b(n)=-1.46+0.5\sin(2\pi fn)$을 이용하여 100번의 독립된 실험을 수행하여 얻은 학습곡선의 결과를 나타낸 것이다. 이때 사용한 주파수는 $f=1/100$을 사용하였다. 수렴 속도는 RLS, D-RLS, NLMS 알고리즘 순으로 우수하고, 수렴 후 MSE 값은 RLS 및 D-RLS 알고리즘이 비슷하고 NLMS 알고리즘가 가장 큼을 알 수 있다. 위 그림에서 D-RLS 알고리즘은 NLMS 알고리즘에 비하여 우수한 성능을 나타냄을 알 수 있다. 시뮬레이션 결과에서 알 수 있듯이 D-RLS 알고리즘은 NLMS 알고리즘 보다 성능이 우수하다. 그러나 colored noise를 만드는 필터의 pole 값에 RLS 알고리즘 보다 영향을 많이 받음을 알 수 있다. 이는 알고리즘 유도할 때 입력 신호들이 상관관계가 없다(uncorrelated)라고 가정하여지만 실제로는 상관관계가 존재하기 때문이다.
결론적으로 D-RLS 알고리즘은 다른 형태의 NLMS 알고리즘의 형태로 보아도 무방할 것으로 생각한다. NLMS의 계산량이 약 4M 이고 D-RLS 알고리즘의 계산량 6M 정도이기 때문에 계산량의 증가는 적으나 성능의 차이는 매우 크다고 할 수 있다.

Ⅳ. 결 론

본 논문은 QR 분해에서 Givens 회전 행렬들에 의하여 입력 신호들의 상관관계가 없어진다는 가정에서 출발하여 QR과 RLS의 관계를 이용하여 RLS 형태의 D-RLS 알고리즘을 개발하였다. 그 계산량은 $O(M)$이다. 위의 가정을 이용한 알고리즘 유도에서 반복시간 $n-1$과 n에서 QR 분해된 데이터 행렬 R이 대각행렬에서 비 대각행렬이 되는 점을 착안하여 반복시간 $n-1$과 n에서 QR 분해된 데이터 행렬 R이 계속 대각행렬이 될 수 있는 방법을 이용하여 알고리즘을 유도하였다. D-RLS은 일반적인 입력 신호 벡터의 자기 상관행렬의 주 대각 요소만 취하여 유도한 RLS 알고리즘 또는 시변 적응 상수를 갖는 LMS 계열의 알고리즘들과 유사한 형태를 취한다. 시변 적응 상수를 갖는 LMS 계열의 알고리즘들은 안정도를 고려하여 초기조건들에 민감하지만 D-RLS 알고리즘은 망각계수 λ와 초기조건 δ만을 정하면 되기 때문에 사용이 편리하다. 시뮬레이션은 FIR 필터의 시스템 식별문제에 적용하여 LMS, NLMS 및 RLS 알고리즘과 제안 알고리즘의 성능을 비교하였다. 수렴특성 비교에서 NLMS 알고리즘보다 우수함을 알 수 있었다. 또한 비선형 시스템인 Volterra 시스템의 식별 문제에 적용하였다.

Kernel이 고정된 경우와 시변으로 하여 컴퓨터 시뮬레이션을 하였다. 그 결과 RLS 알고리즘 보다는 성능이 떨어지지만 NLMS 알고리즘 보다는 우수한 수렴 특성을 나타내었다.

향후 제안한 알고리즘을 직교 입력 벡터 사용하는 알고리즘에 적용할 계획이다.

참 고 문 헌

[1] T. Y. and B. Shahrrava, "Performance of variable step-size LMS algorithms for linear adaptive inverse control systems," Electrical and Computer Engineering, 2005. Canadian Conference on 1-4, pp.755 - 758, May 2005.

[2] M. T. Akhtar, M. Abe and M. Kawamata, "A new variable step size LMS algorithm-based method for improved online secondary path modeling in active noise control systems," Audio, Speech and Language Processing, vool. 14, Issue 2, pp.720 - 726, March 2006.

[3] R. C. Bilcu, P. Kuosmanen and K. Egiazarian, "A transform domain LMS adaptive filter with variable step-size," IEEE Signal Processing Letters, vol. 9, Issue 2, pp.51 - 53 Feb. 2002.

[4] Hyun-Chool Shin, A. H. Sayed, Woo-Jin Song, "Variable step-size NLMS and affine projection algorithms," IEEE, Signal Processing Letters, vol. 11, Issue 2, Part 2, pp.132 - 135, Feb. 2004.

[5] M. T. Morf, T. Kailath and L. Ljung, "Fast algorithm for recursive identification," in proc., 1976 Congress Decision Control, Clearwater Beach, FL, pp.916-92, 1976.

[6] J. M. Cioffi and T. Kailath, "Fast RLS Transversal filters for adaptive filtering," IEEE Trans., Acoust., Speech, Signal Processing, vol. ASSP-32, pp. 304-337, 1984.

[7] W. M. Gentleman and H. T. Kung, "Matrix triangularization by systolic array," in Proc. SPIE int. Soc., Opt. Eng., vol. 298. pp.298-303, 1981.

[8] J. G. McWhiter, "Recursive least squares minimization using a systolic array," in Proc. SPIE int. Soc., Opt. Eng., vol. 431. pp.105-112, 1983.

[9] J. M. Cioffi, "A fast adaptive ROTOR's RLS algorithms," IEEE Trans., Acoust., Speech, Signal Processing, vol. 38, pp.631-653, 1990.

[10] Z. S. Liu, "QR methods of O(N) complexity in adaptive parameter estimation," IEEE Trans, Signal Processing, vol. 43, pp.720-729, 1995.

[11] S. C. Chan, X. X. Yang, , "Improved approximate QR-LS algorithms for adaptive filtering," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 51, Issue 1, pp. 29 - 39, Jan. 2004.

[12] S. Haykin, Adaptive Filter Theory-fouth edition, Upper - Saddle River, New_Jersey 07458: Prentice Hall, 2002.

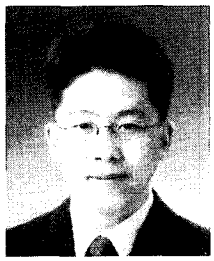
안 봉 만 (Bong-Man Ahn) 정회원

한국통신학회논문지 '07-9 Vol.32, No.8 참조

황 지 원 (JeeWon Hwang) 정회원

한국통신학회논문지 '07-9 Vol.32, No.8 참조

유 정 래 (Jung Rae Ryoo) 정회원



1996년 2월 한국과학기술대학 전기및전자공학과 학사

1998년 2월 한국과학기술원 전기및전자공학과 석사

2004년 2월 한국과학기술원 전자전산학과 박사

2004년~2005년 삼성전자 반도체총괄 책임연구원

2005년~현재 서울산업대학교 제어계측공학과 전임강사

<관심분야> 전자공학, 디지털 신호처리, DSP 기반 신호처리 시스템 구현

조 주 필 (Juphil Cho) 정회원

한국통신학회논문지 '07-9 Vol.32, No.8 참조