Diffusion LMS Algorithm with Data-saved Periodic Combination

이재우, 이한솔, 공준택, 송우진
포항공과대학교 전자전기공학과
{magic00ad, skyst24, gigaleaf, wjsong}@postech.ac.kr

초록: 본 논문에서는 기존 diffusion LMS 알고리즘의 동신량 감소를 위해 combination을 periodically 수행하는 방법을 도입하고, 이로 인해 발생하는 추정값의 oscillation 문제를 해결하기 위한 data-saved periodic combination 방법을 제안한다. 또한 제안된 알고리즘의 성능을 수학적으로 분석하고 실험결과를 통해 기존 동신량 감소 알고리즘들에 비해 제안된 알고리즘이 같은 동신량 감소 대비 더 좋은 추정 성능을 가짐을 확인하였다.

주제어: Distributed estimation, diffusion LMS, periodic combination

I. 서론

본 논문에서는 특정 공간상에 위치한 노드들이 서로 정보를 주고 받으며 이를 이용하여 특정 벡터 파라미터를 추정하는 distributed estimation 문제에 대해 다루고 있다. 특정 공간에 N개의 노드들이 분포되어 있을 때, 노드 k는 매 시간 t마다 scalar값인 desired response \( d_k(t) \) 와 1xM regressor vector \( u_{k,t} \) 를 받는다고 가정한다. 노드 k는 자신의 통신반경 안에 있는 다른 노드들과 서로 정보를 주고받을 수 있고 이러한 노드들을 k의 이웃노드라 부르며 \( N_k \) 라 표기한다. 이러한 상황에서 각 노드들은 자신의 이웃노드들과의 정보 공유를 이용하여 \( d_k(t) \) 와 \( u_{k,t} \) 을 관리하는 특정 벡터 파라미터인 \( w^* \)를 추정하는 것을 목표로 한다.


본 논문에서는 기존의 방식에 비해 간단한 periodic combination이라는 방법을 도입하고 이에 따라 별개의 새로운 문제인 블리너의 형태의 추정값의 oscillation 현상을 해결하기 위한 새로운 방법만을 제안한다. 또한 제안된 알고리즘의 성능을 수학적으로 분석하고 실험결과를 통해 제안된 알고리즘의 우수성을 보인다.

II. Diffusion LMS

본 논문에서는 노드 k가 매 시간 t마다 받는 desired response \( d_k(t) \) 와 regessor vector \( u_{k,t} \) 가 다음과 같은 선형모델을 만족한다고 가정한다.

\[
\hat{d}_k(t) = u_{k,t} w^* + v_k(t).
\]  

여기서 \( v_k(t) \) 는 zero-mean measurement noise를 의미하며 분산은 \( \sigma^2_k \) 이다. \( v_k(t) \) 와 \( u_{k,t} \) 는 모든 \( k,l,i,j \) 에 대해 independent하다고 가정한다.

Diffusion LMS 알고리즘은 adaptation과 combination 두 가지 단계로 이루어져 있다. 우선 adaptation 단계에서는 각 노드들이 생성한 값인 \( d_k(t) \) 와 \( u_{k,t} \) 를 이용하여 각각 추정을 수행한다. 그 다음 combination 단계에서는 adaptation 단계에서 생성된 각각의 추정값들을 이웃노드들과 교환하고, 이를 결합한다. 이러한 combination을 매 시간마다 수행함으로써 결국 각 노드들은 이웃노드들과의 통신만을 통해 전체 네트워크의 정보들을 이용할 수 있게 된다. 본 논문에서는 다루고자 하는 adopt-then-combine (ATC) diffusion LMS는 adaptation 이후 combination을 수행하는 순서를 가지고 있으며 그 업데이트 식은 다음과 같다.

\[
\psi_{k,i} = w_{k,i,t} + \mu_k u_{k,i,t} d_k(t) \quad \text{(Adaptation)}
\]
\[
w_{k,i} = \sum_{l \in N_k} d_k(t) \psi_{l,i} \quad \text{(Combination)}
\]
\[
e_k(t) = d_k(t) - u_{k,t} w_{k,t} \quad \text{(3)}
\]

\( a_{k,t} \) 는 node k가 노드 l의 정보에 추가 combination weight로써 nonnegative의 값을 가지며 다음의 조건을 만족한다.

\[
a_{k,t} = 0 \quad \text{if} \quad l \notin N_k \quad \text{and} \quad \sum_{l \in N_k} a_{k,t} = 1.
\]  

앞서 말했듯이, (2)의 두 반복 단계인 combination 단계는 각 노드들에게 자기 자신의 정보뿐만 아니라 네트워크 전체에 분포되어 있는 다른 노드들의 정보를 함께 이용할 수 있도록 해주는 역할을 수행한다. 이를 통해 diffusion LMS는 combination 단계 없이 adaptation만을 수행하는 stand-alone LMS 알고리즘에 비해 크게 향상된 성능을 가진다는 것이 밝혀져 있다[2].
III. Diffusion LMS with periodic combination

(2)에서의 combination 단계를 통해 추정 성능을 향상시킬 수 있지만, 이는 노드 사이의 정보 교환을 위한 통신량을 필요로 한다. 이러한 통신량을 감소시키기 위해 방법으로써 본 논문에서는 주기적으로 combination을 수행하는 방법과 이에 상관 없이 뚜렷바꾸어 위의 조정값 oscillation 문제를 해결하는 방법을 제안한다. 위 시간마다 adaptation 단계는 항상 수행하되 combination 단계는 p 번마다 한 번씩 주기적으로 수행하는 diffusion LMS 알고리즘의 업데이트 식은 다음과 같이 정리할 수 있다.

\[ \psi_{i,k} = w_{i,k} + \mu_i u_i c_i(i) \]

\[ w_{i,k} = \begin{cases} \sum_{j \in N_k} a_{i,j} \psi_{j,k} & \text{for mod}(i, p) = 0 \\ \psi_{i,k} & \text{otherwise} \end{cases} \quad (5) \]

이와 같이 주기가 p인 combination 단계를 이용하여 되면 기존 diffusion LMS \((p=1)\) 비해 통신량이 \(1/p\)로 감소한다. 하지만 이러한 통신량의 감소는 뚜렷바꾸어의 정확성 이가 확보되는 속도의 저하로 이어지기 때문에 추정 성능이 떨어지는 결과로 이어지게 된다. 그림 1에서 보듯이 주기 \(p\)가 증가함에 따라 추정 성능이 저하된다. 그림 1에서의 \(\gamma\)축은 Network mean square deviation (MSD)로서, 아래와 같은 식으로 표현한다.

\[ \text{Network MSD} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{E} \left[ \left\| w - w_k \right\|^2 \right] \quad (6) \]

또한 steady state에서 뚜렷바꾸어 형태로 MSD가 진동하는 새로운 현상이 나타나게 되는데 이는 periodic combination을 수행함으로써 발생하는 현상이다. 이러한 현상은 combination과 adaptation이 수행 단계의 transient state의 수렴 후의 steady state에서 성능에 미치는 영향이 각각 다르기 때문에 발생하게 된다. 우선 transient state에서는 아직 노드의 주기값인 \(w_{i,k}\)가 중분히 \(w^*\)에 수렴하지 못한 상황이기 때문에 adaptation 단계에서 매 시간마다의 \(d_i(i)\)와 \(u_{i,k}\)를 이용하여 새로운 수렴을 수행하게 된다. Combination 단계 역시 주의의 더 좋은 환경의 노드와의 주기값을 받아 각기 할당으로써 각각의 추정값을 좀 더 향상시키는 효과를 가져온다. 반면, steady state에서는 두 단계가 \(w_{i,k}\)의 수렴성능에 미치는 영향이 달라진다. 이는 adaptation 단계를 통해 도달할 수 있는 steady state에 비해 combination을 추가적으로 수행함으로써 인에서는 steady-state error가 더 낮기 때문에 발생하게 된다. 이는 (stand-alone LMS에 비해 diffusion LMS의 steady-state error가 더 낮다는 사실과 일치한다.) 본 논문에서 제안하는 periodic combination을 이용한 combination 단계를 수행하지 않는 단계에서는 오직 adaptation만을 수행하고 난 이후의 \(w_{i,k}\)를 추가적으로 이용하게 된다. 그림 1에서 Network MSD는 증가하게 되고, p 시간마다 combination을 수행할 때의 network MSD가 크게 감소하게 되는 것이다.

이러한 뚜렷바꾸어 형태의 현상은 steady state에서 각 노드가 이용하는 추정값의 정확도가 주기적으로 변한다는 것을 의미하기 때문에 바람직하지 않은 현상이라고 할 수 있다. 그림 1에서 본 논문에서는 뚜렷바꾸어 형태로 진동하는 network MSD의 가장 낮은 부분으로 향상 추정값을 유지할 수 있는 data-saved periodic combination 방법을 다음과 같이 제안한다.

1. Data-saved periodic combination

기존의 diffusion LMS 알고리즘은 매 시간 \(i\)마다 adaptation과 combination을 한 번씩 수행한다. 예를 들어 \(p=3\)의 주기로 (5)와 같이 periodic combination을 수행한다면 그림 2의 각 부분과 같이 adaptation만 수행하게 되는 2번의 시간이 존재하게 되고 이 시간에 서는 network MSD가 증가하게 되는 것이다. 본 논문에서 이를 방지할 수 있는 한 가지 방법으로 제안하는 것은 \(d_i(i)\)와 \(u_{i,k}\)를 저장해 놓았다가 combination을 수행하는 순간에 오차들을 데이터를 이용하여 한 번의 \(p\) 번의 combination을 동시에 수행하는 것이다. 이는 그림 2의 각 부분에서 확인할 수 있다. 시간 \(i+1\)과 \(i+2\)에서 combination을 수행하지 않아, 이에 정의하는 \(d_i(i)\)와 \(u_{i,k}\)는 저장해 놓았다가 combination을 수행하는 시간인 \(i+3\)에서 한번에 adaptation 3번을 수행한 후 combination을 수행하는 것이다. 이러한 방법을 이용하면
먼 시간 \( i+1 \) 과 \( i+2 \)에서 쓰이는 추정값들이 합성 combination 이후의 값으로 유지되기 때문에 앞서 그림 3에서 전통하는 network MSD의 가장 낮은 값으로 이 동이 성능을 가지고 있다. 또한 점진적으로 같은 수의 adaptation을 수행하기 때문에 같은 수렴속도를 유지할 수 있다. 비록 데이터를 저장할 메모리가 추가적 으로 필요하지만 그 크기가 크지 않고, 이를 이용하면 데이터나 동신 경로 등의 좀더 현실적인 문제와 연관된 통신량을 크게 줄인다는 본 논문의 목표를 달성할 수 있다.

### IV. Performance analysis

본 섹션에서는 제안된 periodic combination을 이용한 diffusion LMS에 대한 수학적 성능 분석을 수행한다. 제안된 알고리즘의 variance relation을 구하고 이를 이용하여 steady-state Network MSD를 계산한다.

우선 주기가 \( p \)라고 했을 때 \( w_{i,k} \)와 \( w_{i+k,p} \) 사이의 관계식은 다음과 같이 표현된다.

\[
\psi_{i,k} = w_{i,k},
\psi_{i+k,p} = \psi_{i,k} + \mu u_{i,k} d_i(i+n) - u_{i+k,p} w_{i+k,p} \quad \text{for } n=1,\ldots, p
\]

\[
w_{i+k,p} = \sum_{h=0}^{N_h} d_k \psi_{i+k,p}.
\]

또한 weight error vector을 \( \psi = \psi - \psi_{i,k} \)로 정의하고, 모든 노드의 weight error vector를 모든 global weight error vector를 다음과 같이 정의한다.

\[
\Psi = \begin{bmatrix} \psi_{i,k} \\ \vdots \\ \psi_{i+k,p} \\ \vdots \\ \psi_{N_i,k} \end{bmatrix}, \quad \dot{\Psi} = \begin{bmatrix} \dot{\psi}_{i,k} \\ \vdots \\ \dot{\psi}_{i+k,p} \\ \vdots \\ \dot{\psi}_{N_i,k} \end{bmatrix}
\]

이러한 global vector를 이용하여 (7)의 식에 matrix equation 형태로 나타내면 다음과 같다.

\[
\Psi_{i+k,p} = A^T \Phi(i+p,i) \Psi - \sum_{m=0}^{c(k)} A^T \Phi(i+p,i+p-m)M g_{i+p-m}
\]

(8)에서 쓰인 matrix들은 각각 아래와 같이 정의된다.

\[
M = \text{diag}\{\mu_1 I, \ldots, \mu_N I\}
\]

\[
g_k = \text{col}\{u_{i,k} v_{i,k}(i), \ldots, u_{i+k,p} v_{i+k,p}(i)\}
\]

\[
R_k = \text{diag}\{R_{i,k}, \ldots, R_{N_i,k}\}
\]

\[
R_{i+k,p} = u_{i+k,p}^T u_{i+k,p}
\]

\[
A = A \otimes I
\]

\( A \) matrix는 combination weight들 \( d_{i,k} \)들의 \( (l,k) \) element로 가지고, \( \otimes \)는 kronecker product 기호로 의미하며, diag\{\}는 원소 안의 값들은 diagonal element 로 가지는 diagonal matrix이므로 col\{\}은 원소 안의 값들을 column vector 형태로 쓰는 것을 의미한다. \( \Phi(i,j) \)는 다음과 같이 정의되는 characteristic function이다.

\[
\begin{align*}
\Phi(i+1,i) &= (I - M R_{i,i}), \quad \Phi(i,i) = I \\
\Phi(i,j) &= \Phi(i,k) \Phi(k,j) \quad \text{for } i > k > j \geq 0
\end{align*}
\]

(8)의 양변에 사용자가 정할 수 있는 hermitian matrix\( \Sigma \)를 이용한 weighted norm을 계산하면 다음과 같은 variance relation을 구할 수 있다.

\[
\begin{align*}
\sum_{m=0}^{c(k)} \text{Tr} E \begin{bmatrix} \sum_{i=1}^{n} A^T \Phi(i+p,i+p-m)M g_{i+p-m} \end{bmatrix}^T \times
\sum_{m=0}^{c(k)} A^T \Phi(i+p,i+p-m)M g_{i+p-m}
\end{align*}
\]

(11) 수식의 두 번째 항은 다음과 같이 그 형태를 바꿀 수 있다.

\[
\sum_{m=0}^{c(k)} \text{Tr} E \begin{bmatrix} \sum_{i=1}^{n} A^T \Phi(i+p,i+p-m)M g_{i+p-m} \end{bmatrix}^T \times
\sum_{m=0}^{c(k)} A^T \Phi(i+p,i+p-m)M g_{i+p-m}
\]

(11) 전체 expectation을 세부 항목들의 expectation들의 곱으로 approximation하면 다음과 같이 표현한다.

\[
\begin{align*}
\sum_{m=0}^{c(k)} \text{Tr} E \begin{bmatrix} \sum_{i=1}^{n} A^T \Phi(i+p,i+p-m)M g_{i+p-m} \end{bmatrix}^T \times
\sum_{m=0}^{c(k)} A^T \Phi(i+p,i+p-m)M g_{i+p-m}
\end{align*}
\]

(11)에서 \( g = E[g, g'] \)이고 \( R = E[R] \)이다. (13)을 vectorization operation을 이용하여 다음과 같이 전개할 수 있다.

\[
\begin{align*}
\sum_{m=0}^{c(k)} \text{Tr} E \begin{bmatrix} \sum_{i=1}^{n} A^T \Phi(i+p,i+p-m)M g_{i+p-m} \end{bmatrix}^T \times
\sum_{m=0}^{c(k)} A^T \Phi(i+p,i+p-m)M g_{i+p-m}
\end{align*}
\]

(11)에서 쓰인 approximation은 small step-size를 가정함으로써 성립된다[21]. Steady state에서는 \( w_{i,k} \)가 수렴하기 때문에, (14)식은 다음과 같이 전개할 수 있다.

\[
\begin{align*}
\sum_{m=0}^{c(k)} \text{vec}\begin{bmatrix} A^T \Phi(i+p,i+p-m)M g_{i+p-m} \end{bmatrix} \sum_{m=0}^{c(k)} A^T \Phi(i+p,i+p-m)\end{align*}
\]

(15)에서 쓰인 approximation은 small step-size를 가정함으로써 성립된다[21]. Steady state에서는 \( w_{i,k} \)가 수렴하기 때문에, (14)식은 다음과 같이 전개할 수 있다.

\[
\begin{align*}
\sum_{m=0}^{c(k)} \text{vec}\begin{bmatrix} A^T \Phi(i+p,i+p-m)M g_{i+p-m} \end{bmatrix} \sum_{m=0}^{c(k)} A^T \Phi(i+p,i+p-m)
\end{align*}
\]

(15)에서 쓰인 approximation은 small step-size를 가정함으로써 성립된다[21]. Steady state에서는 \( w_{i,k} \)가 수렴하기 때문에, (14)식은 다음과 같이 전개할 수 있다.

\[
\begin{align*}
\sum_{m=0}^{c(k)} \text{vec}\begin{bmatrix} A^T \Phi(i+p,i+p-m)M g_{i+p-m} \end{bmatrix} \sum_{m=0}^{c(k)} A^T \Phi(i+p,i+p-m)
\end{align*}
\]
그림 3. 실험 네트워크 환경

그림 4. 제안된 알고리즘과 기존 동신량 감소 알고리즘의 비교

V. 실험결과

실험은 그림 3과 같이 30개의 노드가 있는 네트워크 환경에서 수행되었고 $M = 8$을 이용하였다. Step-size는 $\mu = 0.05$을 이용하였다. 각 번째 실험결과인 그림 4에서는 제안된 periodic combination을 적용한 알고리즘과 기존 동신량 감소 관련 알고리즘의 비교를 보여주고 있다. 모든 알고리즘의 동신량은 기존 diffusion LMS 대비 25%만을 사용하도록 각각의 파라미터를 조정하였다. 같은 양의 동신량 감소가 있을 때, 제안된 알고리즘의 가장 빠르게 수렴하고 가장 낮은 steady-state network MSD를 가질을 확인할 수 있다. 두 번째 실험결과인 그림 5는 동신량 감소 전제의 실험결과와 실제 실험 결과의 차이를 확인할 수 있다.

VI. 결론

본 논문에서는 기존 diffusion LMS 알고리즘의 동신량 감소를 위해 combination을 periodic하게 수행하는 방법을 제안하고 이에 대한 수학적 분석을 수행하였다. 동신량 감소 시 생기는 성능 감소를 보완하기 위해 제안된 알고리즘이 기존 알고리즘들에 비해 더 좋은 성능을 가지는 것을 실험적으로 확인하였다.

Acknowledgement

This research was supported in part by the MSIP (Ministry of Science, ICT & Future Planning), Korea, under the CITRC (Convergence Information Technology Research Center) support program (NIPA-2014-H0401-14-1001) supervised by the NIPA and in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (2012R1A2A2A01011112)

참고문헌


