Signal Processing 114 (2015) 19–23

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Fast communication

# A new approach to introducing a forgetting factor into the normalized least mean squares algorithm

Kiyoshi Nishiyama *

*Department of Electrical Engineering and Computer Science, Faculty of Engineering, Iwate University, 4-3-5, Ueda, Morioka, Japan*

ARTICLE INFO

*Article history:*
Received 14 October 2014
Received in revised form
9 February 2015
Accepted 13 February 2015
Available online 20 February 2015

*Keywords:*
Adaptive filter
Forgetting factor
NLMS algorithm
RLS algorithm
$H_\infty$ filter
System identification

ABSTRACT

The performances of adaptive filtering algorithms are critically controlled by specific tunable parameters. The convergence rate of the normalized least mean squares (NLMS) algorithm may be accelerated by adjusting the step size parameter. The tracking speed of the recursive least squares (RLS) algorithm may be improved by using the forgetting factor, which has not yet been appropriately introduced into the NLMS algorithm. This work aims to successfully introduce the forgetting factor into the NLMS algorithm using an $H_\infty$ theoretical framework developed to create a unified view of adaptive algorithms for recursively identifying the finite impulse response (FIR) filter coefficients. The performances of the forgetting factor NLMS (FFNLMS) algorithm developed here, in the context of several adaptive filtering applications, are evaluated using computer simulations.



## 1. Introduction

The exponential forgetting factor is typically used to accelerate the convergence rate and tracking speed of the recursive least squares (RLS) algorithm in adaptive filtering applications [1–4]. The forgetting factor exponentially decreases the effects of old data on a summation of the weighted squared output errors of an adaptive filter. The forgetting factor has not previously been introduced into the theoretical framework of the normalized least mean squares (NLMS) algorithm because the original NLMS algorithm was heuristically derived. The NLMS algorithm is an extension (modification) of the least mean squares (LMS) algorithm, which approximates solutions to the mean squared output error minimization problem by replacing the expected gradient in the exact solution with the instantaneous gradient. Consequently, it is not straightforward to introduce the forgetting factor into the theoretical framework of the NLMS algorithm. A novel approach would be needed to successfully do so.

In this work, we develop a theoretical approach to introducing the forgetting factor into the NLMS algorithm using the $H_\infty$ framework presented previously by the author for creating a unified view of adaptive finite impulse response (FIR) filtering algorithms [5]. Interestingly, the form of the resultant forgetting factor NLMS (FFNLMS) algorithm is identical to that of a generalized version of the proportionate NLMS (PNLMS) algorithm [6]; however, their meanings with respect to optimality are quite different. The PNLMS algorithm exploits the sparse nature of impulse responses, in which each filter coefficient is updated by an independent upper-bounded step size proportional to the estimated filter coefficient. Similarly, the improved PNLMS (IPNLMS) algorithm [7,8] and the exponentially weighted step size NLMS algorithm [9] were not derived by solving an optimization problem, whereas the FFNLMS algorithm is $H_\infty$−optimal. The performance characteristics of the FFNLMS algorithm are analyzed and verified in comparison with other adaptive algorithms using computer simulations.

The remainder of this paper is organized as follows: Section 2 describes an approach to deriving the FFNLMS

* Tel./fax: +81 19 621 6475.
*E-mail address:* nisiyama@cis.iwate-u.ac.jp

http://dx.doi.org/10.1016/j.sigpro.2015.02.012
0165-1684/

algorithm using the $H_\infty$ framework. Section 3 investigates the performance characteristics of the resulting FFNLMS algorithm in an acoustic system application. Finally, the conclusions are presented in Section 4.

## 2. A forgetting factor NLMS algorithm

### 2.1. A fast recursive solution to the hyper $H_\infty$ filtering problem

The hyper $H_\infty$ filter requires a computational complexity of $\boldsymbol{O}(N^2)$ to obtain the output estimate $\hat{z}_{k|k}=\boldsymbol{H}_k\hat{\boldsymbol{x}}_{k|k}$ for an $N$-dimensional state-space model : $\boldsymbol{x}_{k+1}=\boldsymbol{x}_k+\boldsymbol{G}_k\boldsymbol{w}_k$, $y_k=\boldsymbol{H}_k\boldsymbol{x}_k+v_k$, $z_k=\boldsymbol{H}_k\boldsymbol{x}_k$, where $y_k$ is the observed output of an unknown system [5]. Fortunately, a fast algorithm for hyper $H_\infty$ filtering, the fast $H_\infty$ filter (FHF) [10], may be applied without the Riccati recursion of $\hat{\boldsymbol{\Sigma}}_{k|k-1}$ if the $1\times N$ observation matrix $\boldsymbol{H}_k$ is characterized by a shift property such that $H_k(i)=H_{k-1}(i-1)$ and $H_k(1)=u_k$, where $H_k(i)$ is the $i$-th element of $\boldsymbol{H}_k$ at time index $k$, and $u_k$ is the input to the unknown system. The level-$\gamma_f$ FHF algorithm can be recursively calculated at a computational complexity of $\boldsymbol{O}(N)$ per time step, according to

$$\hat{\boldsymbol{x}}_{k|k}=\hat{\boldsymbol{x}}_{k-1|k-1}+\boldsymbol{K}_{s,k}(y_k-\boldsymbol{H}_k\hat{\boldsymbol{x}}_{k-1|k-1}),\quad \hat{z}_{k|k}=\boldsymbol{H}_k\hat{\boldsymbol{x}}_{k|k} \tag{1}$$

$$\boldsymbol{K}_{s,k}=\frac{\boldsymbol{K}_k(:,1)}{1+\gamma_f^{-2}\boldsymbol{H}_k\boldsymbol{K}_k(:,1)},\quad \boldsymbol{K}_k=\boldsymbol{m}_k-\boldsymbol{D}_k\boldsymbol{\mu}_k \tag{2}$$

$$\boldsymbol{D}_k=\frac{\boldsymbol{D}_{k-1}-\boldsymbol{m}_k\boldsymbol{W}\boldsymbol{\eta}_k}{1-\boldsymbol{\mu}_k\boldsymbol{W}\boldsymbol{\eta}_k},\quad \boldsymbol{\eta}_k=\boldsymbol{c}_{k-N}+\boldsymbol{C}_k\boldsymbol{D}_{k-1} \tag{3}$$

$$\begin{bmatrix}\boldsymbol{m}_k\\ \boldsymbol{\mu}_k\end{bmatrix}=\begin{bmatrix}S_k^{-1}\boldsymbol{e}_k^T\\ \boldsymbol{K}_{k-1}+\boldsymbol{A}_kS_k^{-1}\boldsymbol{e}_k^T\end{bmatrix},\quad \boldsymbol{W}=\begin{bmatrix}1 & 0\\ 0 & -\gamma_f^{-2}\end{bmatrix} \tag{4}$$

$$S_k=\rho S_{k-1}+\boldsymbol{e}_k^T\boldsymbol{W}\tilde{\boldsymbol{e}}_k,\quad \boldsymbol{e}_k=\boldsymbol{c}_k+\boldsymbol{C}_{k-1}\boldsymbol{A}_k \tag{5}$$

$$\boldsymbol{A}_k=\boldsymbol{A}_{k-1}-\boldsymbol{K}_{k-1}\boldsymbol{W}\tilde{\boldsymbol{e}}_k,\quad \tilde{\boldsymbol{e}}_k=\boldsymbol{c}_k+\boldsymbol{C}_{k-1}\boldsymbol{A}_{k-1} \tag{6}$$

where the $2\times N$ matrix $\boldsymbol{C}_k$ consists of $\boldsymbol{H}_k$ as $\boldsymbol{C}_k=[\boldsymbol{H}_k^T,\boldsymbol{H}_k^T]^T$ and $\boldsymbol{c}_k\in\mathcal{R}^{2\times1}$ is the first row of $\boldsymbol{C}_k=[\boldsymbol{c}_k,\ldots,\boldsymbol{c}_{k-N+1}]$, assuming that $\boldsymbol{c}_{k-i}=\boldsymbol{0}_{2\times1}$ for $k-i<0$. The forgetting factor $\rho$ is typically selected to be $0<\rho=1-\chi(\gamma_f)\le1$ for $\gamma_f>1$. The recursions are initialized using $\boldsymbol{K}_0=\boldsymbol{0}_{N\times2}$, $\boldsymbol{A}_0=\boldsymbol{0}$, $S_0=1/\varepsilon_0$, $\boldsymbol{D}_0=\boldsymbol{0}$, $\hat{\boldsymbol{x}}_{0|0}=\boldsymbol{0}$ where $\varepsilon_0$ is set to be a relatively large positive number, $\boldsymbol{0}$ denotes the zero vector, and $\boldsymbol{0}_{n\times m}$ stands for the $n\times m$ zero matrix.

The computational burden of the FHF algorithm is comparable to the burdens of the fast Kalman filter (FKF) and the fast transversal filter (FTF) [1]. Note that for $\gamma_f=\infty$, the FHF algorithm coincides with the FKF algorithm.

### 2.2. An alternative form of the FHF recursion

The gain matrix $\boldsymbol{K}_k$ in the FHF can be expressed in a recursive form as

$$\begin{aligned}\boldsymbol{K}_k&=\boldsymbol{m}_k-\boldsymbol{D}_k\boldsymbol{\mu}_k=[\boldsymbol{I}_{N\times N}-\boldsymbol{D}_k]\begin{bmatrix}\boldsymbol{m}_k\\ \boldsymbol{\mu}_k\end{bmatrix}\\&=[\boldsymbol{I}_{N\times N}-\boldsymbol{D}_k]\left\{\begin{bmatrix}\boldsymbol{0}_{1\times2}\\ \boldsymbol{K}_{k-1}\end{bmatrix}+\frac{1}{S_k}\begin{bmatrix}1\\ \boldsymbol{A}_k\end{bmatrix}\boldsymbol{e}_k^T\right\}\\&=[\boldsymbol{I}_{N\times N}-\boldsymbol{D}_k]\begin{bmatrix}\boldsymbol{0}_{1\times2}\\ \boldsymbol{K}_{k-1}\end{bmatrix}+\frac{1}{S_k}[\boldsymbol{I}_{N\times N}-\boldsymbol{D}_k]\begin{bmatrix}1\\ \boldsymbol{A}_k\end{bmatrix}\boldsymbol{e}_k^T.\end{aligned} \tag{7}$$

Consequently, the FHF algorithm may be rewritten as

$$\begin{aligned}&\hat{\boldsymbol{x}}_{k|k}=\hat{\boldsymbol{x}}_{k-1|k-1}+\boldsymbol{K}_{s,k}(y_k-\boldsymbol{H}_k\hat{\boldsymbol{x}}_{k-1|k-1}),\\&\boldsymbol{K}_{s,k}=\frac{\boldsymbol{K}_k(:,1)}{1+\gamma_f^{-2}\boldsymbol{H}_k\boldsymbol{K}_k(:,1)}\end{aligned} \tag{8}$$

$$\boldsymbol{K}_k=[\boldsymbol{I}_{N\times N}-\boldsymbol{D}_k]\begin{bmatrix}\boldsymbol{0}_{1\times2}\\ \boldsymbol{K}_{k-1}\end{bmatrix}+\frac{1}{S_k}[\boldsymbol{I}_{N\times N}-\boldsymbol{D}_k]\begin{bmatrix}1\\ \boldsymbol{A}_k\end{bmatrix}\boldsymbol{e}_k^T \tag{9}$$

where the recursive variables $\boldsymbol{A}_k$, $S_k$, and $\boldsymbol{D}_k$ are updated as they are in the original FHF algorithm. Note that $\boldsymbol{K}_k(:,1)$ denotes the first column vector of $\boldsymbol{K}_k$.

### 2.3. Algorithm derivation

As shown in Fig. 1 of [5], the recursive solution at the point $(\gamma_f,\sigma_w^2)=(1,0)$ on the $\gamma_f$-$\sigma_w^2$ plane coincided with the NLMS algorithm with a step size of $\mu=1$. The NLMS algorithm, therefore, is strictly $H_\infty$−optimal, that is consistent with the conclusions of [11].

We next consider the solutions that satisfy $\sigma_w^2>0$ on the line $\gamma_f=1$, where $\sigma_w^2$ is defined as the variance of $w_k$ in the state error-dependent noise $(\hat{\boldsymbol{x}}_{k|k}-\boldsymbol{x}_k)w_k$. The state error-dependent noise is equivalent to $\boldsymbol{G}_k\boldsymbol{w}_k$ whose covariance matrix satisfies $\boldsymbol{G}_k\boldsymbol{\Sigma}_{\boldsymbol{w}_k}\boldsymbol{G}_k^T=\sigma_w^2\hat{\boldsymbol{\Sigma}}_{k|k}$, which functions as a forgetting factor [5]. Applying $\gamma_f=1$ to the FHF recursion described in (2) to (6), after decoupling the relationship between $\sigma_w^2$ and $\gamma_f$, we find that the recursive variables $\boldsymbol{A}_k$ and $\boldsymbol{D}_k$ become zero, and $S_k$ is given by

$$S_k=\rho S_{k-1}=\rho^kS_0,\quad S_0=\varepsilon_0^{-1},\quad \rho=\frac{1}{1+\sigma_w^2}. \tag{10}$$

The gain matrix recursion of (9) is then reduced to

$$\begin{aligned}\boldsymbol{K}_k&=[\boldsymbol{I}_{N\times N}\ \boldsymbol{0}]\begin{bmatrix}\boldsymbol{0}_{1\times2}\\ \boldsymbol{K}_{k-1}\end{bmatrix}+\frac{1}{S_k}[\boldsymbol{I}_{N\times N}\ \boldsymbol{0}]\begin{bmatrix}1\\ \boldsymbol{0}\end{bmatrix}\boldsymbol{c}_k^T\\&=\begin{bmatrix}0 & \cdots & \cdots & 0 & 0\\ 1 & \ddots & & \vdots & \vdots\\ 0 & \ddots & \ddots & \vdots & \vdots\\ \vdots & \ddots & \ddots & 0 & \vdots\\ 0 & \cdots & 0 & 1 & 0\end{bmatrix}\boldsymbol{K}_{k-1}+\frac{1}{S_k}\begin{bmatrix}\boldsymbol{c}_k^T\\ 0\\ \vdots\\ \vdots\\ 0\end{bmatrix}.\end{aligned} \tag{11}$$

The nonrecursive form of (11) may be given, after certain calculations, by

$$\boldsymbol{K}_k=\begin{bmatrix}\boldsymbol{c}_k^T/S_k\\ \boldsymbol{c}_{k-1}^T/S_{k-1}\\ \vdots\\ \boldsymbol{c}_{k-N+1}^T/S_{k-N+1}\end{bmatrix}=\frac{1}{S_k}\begin{bmatrix}\boldsymbol{c}_k^T\\ \rho\boldsymbol{c}_{k-1}^T\\ \vdots\\ \rho^{N-1}\boldsymbol{c}_{k-N+1}^T\end{bmatrix}. \tag{12}$$

Substituting this result into (2) and setting $\gamma_f=1$, we obtain an adaptive algorithm:

$$\hat{\boldsymbol{x}}_{k|k}=\hat{\boldsymbol{x}}_{k-1|k-1}+\frac{\boldsymbol{\Lambda}\boldsymbol{H}_k^T}{\|\boldsymbol{H}_k^T\|_{\boldsymbol{\Lambda}}^2+S_k}(y_k-\boldsymbol{H}_k\hat{\boldsymbol{x}}_{k-1|k-1}) \tag{13}$$

where $\|\boldsymbol{H}_k^T\|_{\boldsymbol{\Lambda}}^2=\boldsymbol{H}_k\boldsymbol{\Lambda}\boldsymbol{H}_k^T$ is the squared norm of the vector $\boldsymbol{H}_k^T$ weighted with $\boldsymbol{\Lambda}$, and $\boldsymbol{\Lambda}$ is the diagonal matrix of the elements $1,\rho,\ldots$, and $\rho^{(N-1)}$. This algorithm is referred to as a forgetting