SERIES-CASCADE NONLINEAR ADAPTIVE FILTERS

V. Hegde C. Radhakrishnan, D. Krusienski, and W. K Jenkins
Department of Electrical Engineering
The Pennsylvania State University
University Park, PA

Abstract – It is known that certain classes of nonlinear systems can be represented by one of three cascade models: i) a linear filter followed by a memoryless nonlinearity (Wiener model), ii) a memoryless nonlinearity followed by a linear filter (Hammerstein model), or iii) a linear filter, a memoryless nonlinearity, and a second linear filter (LNL model). In this paper we consider LNL adaptive systems with an FIR linear system at the input stage and a FIR linear system at the output stage. Then combining the linear input stage and the memoryless nonlinear stage of the LNL model is considered, resulting in the series-cascade of a Wiener system with a linear output stage. Adaptive algorithms are derived for these structures and experimental examples are shown to illustrate their performance.

1. Introduction

Many nonlinear systems can be represented using one of three models shown in Figure 1 [1]. If the memory size of the linear component in the Wiener model or the Hammerstein model is small it may be desirable to model them using Volterra filters, by virtue of its modularity. Similarly, depending on the memory size of the linear components in the LNL model, it may be desirable to represent it by a single Volterra filter, or a cascade of a Volterra filter followed by linear system, or a cascade of a linear system followed by a Volterra filter.

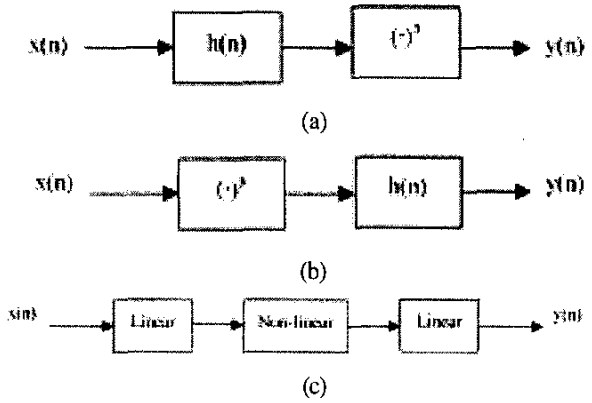

Figure 1. (a) Wiener model, (b) Hammerstein model, and (c) LNL model.

In this work, we consider LNL systems with FIR linear system at the input stage and a FIR at the output stage. In particular, we consider the two models shown in Figure 2

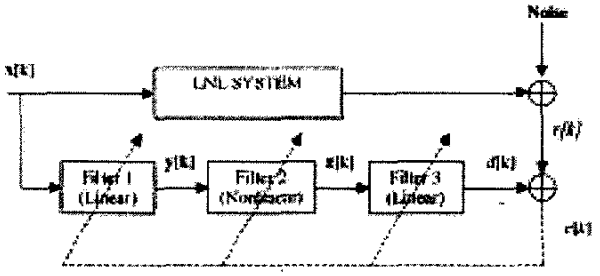

Figure 2. Adaptive LNL filter structure.

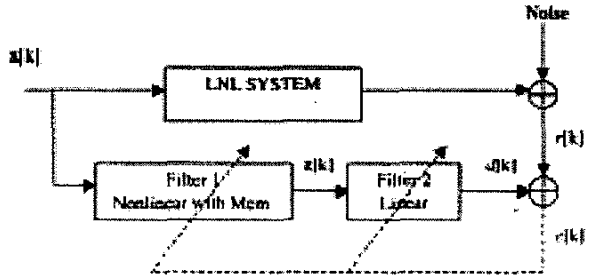

Figure 3. Cascade of Volterra and FIR filter structures.

and 3 to represent such a system. The nonlinear filter in Figure 2 can be implemented with a memoryless Polynomial filter. Filter-1 of Figure 3 is a nonlinear filter with memory, which can be implemented with a Volterra filter . The configurations considered can be categorized into two ways; one particular case in each of these categories is developed in this paper:

Hammerstein Model + Linear system: Volterra filter + FIR
LNL Model: FIR + memoryless Polynomial filter + FIR

A disadvantage of series-cascade adaptive architectures for system identification is that there is no reference signal at the output of each stage in the cascade. Hence the adaptation of the filter modules in the cascade prior to the the output stage have to rely on the joint error signal $e[n]$.

2. Joint Adaptation Schemes

In this section joint adaptation schemes for the cascade structures described above are derived. In the experiments presented in Section 3 the normalized least mean squares (NLMS) method is used for adaptation. The NLMS algorithm requires the derivative of the error-squared with respect to the filter coefficients for updating the tap weights. These derivatives for each section of the cascade structure are derived in the following subsections, and the tap-update

0-7803-7523-8/02/$17.00 ©2002 IEEE

equations for the joint adaptation scheme based on NLMS algorithm are developed.

2.1 Cascaded Volterra and FIR filter Structure

Let **v** and **w** be the coefficient vectors, and **y** and **z** be the input vectors of the Volterra filter and the FIR filter respectively (see figure 6). The outputs of the two filters $z[k]$ and $d[k]$ are given by equations (1) and (2), respectively:

$$z[k] = \mathbf{v}^T[k]\mathbf{y}[k], \quad (1)$$

$$d[k] = \mathbf{w}^T[k]\mathbf{z}[k], \quad (2)$$

where $\mathbf{y}[k] = [\, y_0[k], y_1[k], \ldots, y_{N_v-1}[k] \,]^T$,

and $\mathbf{z}[k] = [\, z[k], z[k-1], \ldots, z[k-M_2+1] \,]^T$.

If $r[k]$ is the reference signal for the cascade, then the error, e(n) is given by:

$$e[k] = r[k] - d[k], \quad (3)$$

The gradients ∇_w and ∇_v of e^2, at instant k with respect to **w** and **v** can be calculated according to:

$$\nabla_w e^2[k] = -2e[k]\, \mathbf{z}[k], \quad (4)$$

$$\nabla_v e^2[k] = -2e[k]\, (\nabla_v \mathbf{z}[k])\mathbf{w}[k]. \quad (5)$$

Also, $z[n] = \mathbf{v}^T[n]\, \mathbf{y}[n]$ and $\nabla_v z[n] = \mathbf{y}[n]$.

Combining the above relations results in:

$$\mathbf{Y}[k] = \nabla_v \mathbf{z}[k] = [\mathbf{y}[k], \mathbf{y}[k-1], \ldots, \mathbf{y}[k-M_2+1]\,]. \quad (6)$$

Y is a $N_v \times M_2$ matrix. Defining vector $\mathbf{q}[k] = \mathbf{Y}[k]\, \mathbf{w}[k]$ and using it in equations (5) and (6) we get,

$$\nabla_v e^2[k] = -2e[k]\, \mathbf{q}[k]. \quad (7)$$

Table 1 summarizes the joint NLMS adaptation algorithm for the two sections. The computational requirements for this cascade structure are summarized as follows:

Number of tap-weights:	N_v+M_2
Memory units:	N_vM_2
Additions:	$2N_v+3M_2+N_vM_2-1$
Multiplications:	$M_v+3N_v+4M_2+N_vM_2+4$

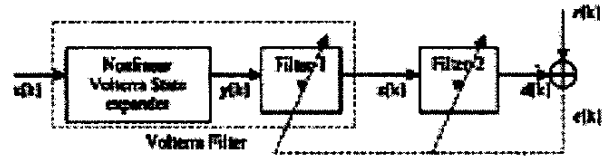

Figure 4. Volterra Filter + FIR filter cascade strructure.

2.2 The General Cascaded LNL Structure

Let **u** and **x**, be the coefficient vector and the input vector, respectively, for FIR filter-1. Similarly define **v** and **y** for the Polynomial filter, and **w** and **z** for FIR filter-2 (see figure 5). The outputs of the three filters $y[k]$, $z[k]$ and $d[k]$ are given by equations (8), (9) and (10), respectively:

$$y[k] = \mathbf{u}^T[k]\, \mathbf{x}[k], \quad (8)$$

$$z[k] = \mathbf{v}^T[k]\, \mathbf{y}[k], \quad (9)$$

$$d[k] = \mathbf{w}^T[k]\, \mathbf{z}[k], \quad (10)$$

where $\mathbf{x}[k] = [\, x[k], x[k-1], \ldots, x[k-M_1+1] \,]^T$,

$\mathbf{y}[k] = [\, y[k], y^2[k], \ldots, y^N[k] \,]^T$,

and $\mathbf{z}[k] = [\, z[k], z[k-1], \ldots, z[k-M_2+1] \,]^T$,

If $r[k]$ is the reference signal for the cascade, then error, e is given by:

$$e[k] = r[k] - d[k]. \quad (11)$$

The gradient ∇_w, ∇_v and ∇_u of e^2, at instant k, with respect to **w**, **v** and **u** respectively can be calculated according to:

$$\nabla_w e^2[k] = -2e[k]\, \mathbf{z}[k], \quad (12)$$

$$\nabla_v e^2[k] = -2e[k]\, (\nabla_v \mathbf{z}[k])\mathbf{w}[k], \quad (13)$$

$$\nabla_u e^2[k] = -2e[k]\, (\nabla_u \mathbf{z}[k])\mathbf{w}[k]. \quad (14)$$

The procedure for calculating $\nabla_v e^2$ is the same as that used in Section 3.1, which given by:

$$\nabla_v e^2[k] = -2e[k]\, \mathbf{q}[k], \quad (15)$$

where, **q** is calculated using Equations (16) – (18):

$$z[n] = \mathbf{v}^T[n]\, \mathbf{y}[n], \quad (16)$$

$$\nabla_v z[n] = \mathbf{y}[n], \quad (17)$$

where $\mathbf{Y}[k] = \nabla_v \mathbf{z}[k] = [\mathbf{y}[k], \mathbf{y}[k-1], \ldots, \mathbf{y}[k-M_2+1]]$,

and $\mathbf{q}[k] = \mathbf{Y}[k]\, \mathbf{w}[k]$. (18)

To calculate $\nabla_u e^2$ we start by defining vector $\mathbf{a}[n]$ given by equation (19):

$$\mathbf{a}[n] = [1,\ 2y[n],\ 3y^2[n], \ldots\ldots,\ Ny^{N-1}[n]\,]^T. \quad (19)$$

We have $\nabla_u\, y^i[n] = i.y^{i-1}[n]\, \nabla_u\, y[n]$. Using this and equations (16) and (19) we get,

$$\nabla_u z[n] = \mathbf{v}^T[n]\, (\mathbf{a}[n]\, \nabla_u y[n]) = (\mathbf{v}^T[n]\, \mathbf{a}[n])\, \nabla_u y[n]. \quad (20)$$

Defining scalar $b[n] = \mathbf{v}^T[n].\ \mathbf{a}[n]$ and using it in equation (20) we get,

$$\nabla_u z[n] = b[n]\, \nabla_u y[n]. \quad (21)$$

We have $\nabla_u y[n] = \mathbf{x}[n]$ from equation (8). Using this in equation (21) results in,

$$\nabla_u z[n] = b[n]\, \mathbf{x}[n]. \quad (22)$$

Defining a $M_1 \times M_2$ matrix,

$\mathbf{X}[k] = [b[k]\mathbf{x}[k],\ b[k-1]\mathbf{x}[k-1], \ldots\ldots,\ b[k-M_2+1]\mathbf{x}[k-M_2+1]]$,

and using this in equation (22) and we get,

$$\nabla_u z[k] = \mathbf{X}[k]. \quad (23)$$

Defining a M_1 length vector, $\mathbf{p}[k] = \mathbf{X}[k].\mathbf{w}[k]$ and using equations (14) and (23) we get,

$$\nabla_u e^2[k] = -2e[k]\ \mathbf{p}[k]. \qquad (25)$$

Table 2 summarizes the joint NLMS adaptation of the three sections. The computatinal requirements for this cascade structue are as follows:

Number of tap-weights:	M_1+N+M_2
Memory units:	$(M_1+N)M_2$
Additions:	$2M_1+3N+3M_2+ M_1M_2+NM_2-3$
Multiplications:	$4M_1+6N+3M_2+M_1M_2+NM_2+4$

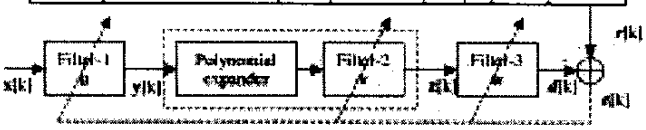

Figure 5. Cascade of FIR, memoryless polynomial and FIR filters.

3. Experimental Results

The cascade structures described in Section 2 were tested for the identification of a nonlinear system. Figure 6 shows the model of the echo path that was used for the simulations, in which the loudspeaker was modeled as a Wiener model and the acoustic echo path by a linear filter.

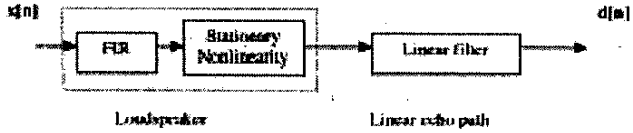

Figure 6. Model of the echo path.

Modeling the loudspeaker: The loudspeaker was modeled with a FIR filter with memory length 8 in series cascade with a memoryless nonlinearity of order 5 [2]. The first coefficient of the FIR filter was kept at value 1 and remaining 7 coefficients were generated randomly and were kept small, so that the gain due to the memory of this FIR filter is about 25%. This makes sure that if the magnitude of the input to the loudspeaker is within 0.8, i.e., the magnitude of the input to the static nonlinearity is within 1.0. The amplifier was modeled by the nonlinear function $f(x) = x - 0.5x^3 + 0.02x^5$, which approximates a hyperbolic tangent. The amplifier is nearly linear for an input [-0.3, 0.3], but starts saturating outside this range.

Modeling the linear part of the echo path: The envelope of the impulse response of the room typically has a peak, and decays exponentially on either side. Also, it has been noted that an echo path transfer function usually exhibits an all-pass characteristic. For all simulations in an FIR filter of memory length 64 was used. A small random number was added to each coefficient. For all simulations an all-pass filter with two complex conjugate poles at $0.9e^{\pm j60}$ and two zeros at $1.1111e^{\pm j60}$ was used [3].

A babble signal was used as a test signal for the simulations presented here. Since the main purpose is to identify the nonlinear components in the echo path, the instantaneous magnitude of the input signal was kept in the range [0.3,0.8], so that the loudspeaker operates in its nonlinear region. A Gaussian noise of variance 0.001 was added to the echo signal, corresponding to a SNR of 21 dB for the echo signal.

Experiment 1: Since he lengths of the linear section in the loudspeaker and the echo path are M_1=8 and M_2=64, the AEC should have a length of at least $M = M_1+M_2-1 = 71$. As a baseline experiment an FIR filter of length 71 was used for acoustic echo cancellation. during the first 20,000 iterations the babble input was de-amplified by 0.375 so that the input remained in the range [-0.3,0.3], and the loudspeaker operated in its linear region. For the next 20,000 iterations the de-amplification was removed and the loudspeaker was driven into its nonlinear region. The echo return loss (ERLE) [3] plot for this simulation is shown in figure 7. The FIR filter achieved an ERLE of 22.5 dB in the linear region. In the nonlinear region the ERLE is only 17 dB, which shows that failure t compensate for the nonlinearity results in a significant loss of performance.

Experiment 2: Next the series cascade of a Volterra filter and an FIR was tested. The Volterra filter used had a memory of $M_1 = 8$ and order of nonlinearity 3, thereby requiring 164 filter coefficients The length of the FIR filter was set to $M_2 = 64$. The ERLE plot of this filter is shown figure 8. This cascade structure converged to 21 dB after 30,000 iterations, which is still 1.5 dB lower than the ERLE obtained in the linear region.

Experiment 3: LNL cascade structure was used for echo cancellation. The FIR filters at the input the output stages had lengths of $M_1 = 8$ and $M_2 = 64$. The Polynomial filter had a nonlinearity of order N = 5. The total number of taps required for this filter is 77. The ERLE plot of this filter is shown figure 9. This cascade structure converged to 20 dB after 30,000 iterations, a level that is approximately 2.5 dB lower than the ERLE obtained in the linear region. This filter was able to account for 3 dB of the 5.5 dB nonlinearity observed in Experiment 1.

A variation of Experiment 3 is presented in figure 10, in which the same structure is trained with a backpropagation algorithm commonly used in neural networks [4]. A comparison of he training characteristics of figures 9 and 10 shows remarkable similarities. However, the back-propagation algorithm has the potential to reduce computational complexity. The application of the backpropagation algorithm in cascaded nonlinear adaptive filter structures is a subject of current research.

References

[1] A. E. Nordsjo and L. H. Zetterberg, Identification of certain time-varying nonlinear Wiener and Hammerstein systems, *IEEE Trans.on Sig. Proc.*, VOL 49, NO. 3, pp 577-592, March 2001.

[2] F. X. Y. Gao and W. M. Snelgrove., "Adaptive Linearization of Loudspeaker," Proc. ICASSP'91, pp. 3589-3592.

[3] H. Fan, and W. K. Jenkins, "An Investigation of an Adaptive IIR Echo Canceller: Advantages and Problems", *IEEE Transactions on Acoustic, Speech, and Signal Processing*, Vol. 36, No. 12, December 1988.

[4] Bose N., Liang P., *Neural Networks: Graphs and Algorithms*, McGraw-Hill Book Company, N.Y., 1996.

Table 1. Joint NLMS update for Volterra filter + FIR filter cascade

	Operation	Additions	Multiplications
1.	$y[k] = [\, y_0[k], y_1[k], \ldots, y_{N_v-1}[k] \,]^T$	-	M_v
2.	$z[k] = v^T[k].y[k]$	N_v-1	N_v
3.	$z[k] = [\, z[k], z[k-1], \ldots, z[k-M+1] \,]^T$	-	-
4.	$d[k] = w^T[k].z[k]$	M_2-1	M_2
5.	$e[k] = r[k] - d[k]$	1	-
6.	$Y[k] = [\, y[k], y[k-1], \ldots, y[k-M+1] \,]$	-	-
7.	$q[k] = Y[k].w[k]$	$(M_2-1)N_v$	M_2N_v
8.	$v[k+1] = v[k] + (\alpha_v / (\|q[k]\|^2 + \delta)).q[k].e[k]$	$2N_v$	$2N_v+2$
9.	$w[k+1] = w[k] + (\alpha_w / (\|z[k]\|^2 + \delta)).z[k].e[k]$	$2\,M_2$	$2\,M_2+2$
10.			

Table 2. Joint NLMS update for FIR +memoryless polynomial filter + FIR cascade

	Operation	Additions	Multiplications
1.	$x[k] = [\, x[k], x[k-1], \ldots, x[k-M_1+1] \,]^T$	-	-
2.	$y[k] = u^T[k].x[k]$	M_1-1	M_1
3.	$y[k] = [\, y[k], y[k]^2, \ldots, y[k]^N \,]^T$	-	N-1
4.	$z[k] = v^T[k].y[k]$	N-1	N
5.	$z[k] = [\, z[k], z[k-1], \ldots, z[k-M_2+1] \,]^T$	-	-
6.	$d[k] = w^T[k].z[k]$	M_2-1	M_2
7.	$e[k] = r[k] - d[k]$	1	-
8.	$a[k] = [1, 2\,y[k], 3y[k]^2, \ldots, Ny[k]^{N-1}]$	-	N-1
9.	$b[k] = v^T[k].a[k]$	N-1	N
10.	$x_n[k] = b[k].x[k]$	-	M_1
11.	$X[k] = [x_n[k], x_n[k-1], \ldots, x_n[k-M_2+1]]$	-	-
12.	$p[k] = X[k].w[k]$	$M_1(M_2-1)$	M_1M_2
13.	$Y[k] = [\, y[k], y[k-1], \ldots, y[k-M_2+1] \,]$	-	-
14.	$q[k] = Y[k].w[k]$	$N(M_2-1)$	NM_2
15.	$u[k+1] = u[k] + (\alpha_u / (\|p[k]\|^2 + \delta)).p[k].e[k]$	$2M_1$	$2M_1+2$
16.	$v[k+1] = v[k] + (\alpha_v / (\|q[k]\|^2 + \delta)).q[k].e[k]$	2N	2N+2
17.	$w[k+1] = w[k] + (\alpha_w / (\|z[k]\|^2 + \delta)).z[k].e[k]$	$2M_2$	$2M_2+2$

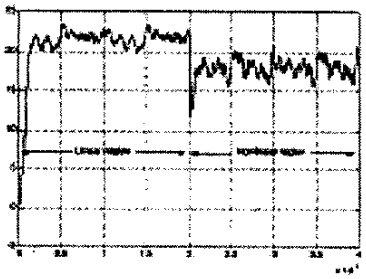

Figure 7. ERLE of FIR-AEC with the loudspeaker operated in its linear and the nonlinear regions.

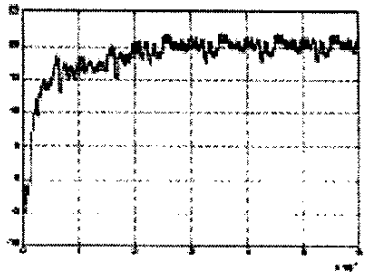

Figure 9. ERLE of cascade of FIR + memoryless polynomial + FIR.

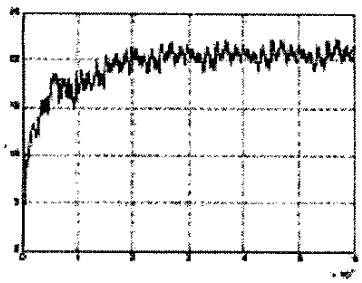

Figure 8. ERLE for cascade of Volterra and FIR filters.

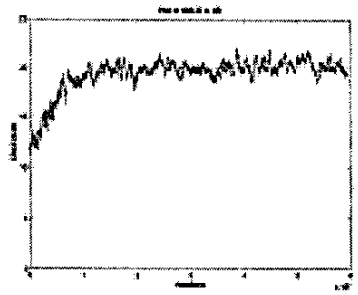

Figure 10. Neural network backpropagation training algorithm.