Volterra Series and Nonlinear Adaptive Filters

Traian Abrudan

tabrudan@wooster.hut.fi.

30.10.2003

Helsinki University of Technology

Signal Processing Laboratory

P.O.Box 3000, 02015-FIN

Outline

1. Introduction
2. Linear vs. Nonlinear Adaptive Filtering
3. Volterra Series Expansion for Continuous/ Discrete Time Nonlinear Systems
4. Volterra Filters in Frequency Domain
5. Time Varying Systems
6. Nonlinear Adaptive Filtering
7. Applications: LMS and RLS Volterra Filters
8. Summary and Conclusions

1. Introduction

Motivation and advantages of the Volterra model:

- ★ The model is very popular and has developed the identity of its own in the last few years
- ★ It is attractive from the mathematical point of view
- ★ It fits a large class of nonlinear systems
- ★ The LMS and RLS adaptive algorithms are suitable for practical implementation

2. Linear vs. Nonlinear Adaptive Filtering

- The obvious advantage of *linear adaptive filters** is their inherent *simplicity*
- Linear filters are known to be optimal if the noise is Gaussian
- In many applications the linear Gaussian model is not adequate anymore:
  - Signal companding
  - Amplifier saturation
  - Multiplicative interaction between Gaussian signals
  - High data rate transmissions (e.g. copper line, satellite links)
  - Biological signals
- In this case the performance of linear filters may become unacceptable (e.g. the BER)

3.1 Volterra Series Expansion of a Discrete Time Nonlinear System

- The Volterra series model is the most widely used model in nonlinear adaptive filtering
- The Volterra series expansion can be seen as a Taylor series expansion with memory.
- A nonlinear continuous function $y = f(x)$ can be expanded to a Taylor series, at $x = x_0$:

$$f(x) = \sum_{l=0}^{\infty} \underbrace{\frac{1}{l!} \frac{\partial^l f(x)}{\partial x^l} \bigg|_{x=x_0}}_{a_l} (x - x_0)^l = \sum_{l=0}^{\infty} a_l (x - x_0)^l$$

3.1 Volterra Series Expansion of a Discrete Time Nonlinear System

- It consists consists of a nonrecursive series in which the output signal is related to the input signal as follows:

$$
\begin{aligned}
y(k) \quad = \quad & w_{o0} + \sum_{l_1=0}^{\infty} w_{o1}(l_1)x(k-l_1) && \text{A constant } + \text{ Linear filter} \\
+ \quad & \sum_{l_1=0}^{\infty}\sum_{l_2=0}^{\infty} w_{o2}(l_1,l_2)x(k-l_1)x(k-l_2) && \text{Higher} - \text{order convolutions} \\
+ \quad & \sum_{l_1=0}^{\infty}\sum_{l_2=0}^{\infty}\sum_{l_3=0}^{\infty} w_{o3}(l_1,l_2,l_3)x(k-l_1)x(k-l_2)x(k-l_3) + \ldots \\
+ \quad & \sum_{l_1=0}^{\infty}\sum_{l_2=0}^{\infty}\cdots\sum_{l_p=0}^{\infty} w_{op}(l_1,l_2,\ldots l_p)x(k-l_1)x(k-l_2)\ldots x(k-l_p) + \ldots
\end{aligned}
$$

The model is attractive in because the expansion is a *linear combination of nonlinear functions of the input signal*

3.1 Volterra Series Expansion of a Discrete Time Nonlinear System

- The coefficients $w_{op}(l_1, l_2, \ldots l_p)$ are the coefficients of a nonlinear combiner based on Volterra series, and called the Volterra series kernels (symmetric).
- The Volterra series expansion generalizes the Taylor series:

$$y(k) = \sum_{p=0}^{P} \mathcal{H}_p[x(k)]$$

where the terms are:

$$\mathcal{H}_p[x(k)] = \sum_{l_1=0}^{L-1} \sum_{l_2=0}^{L-1} \ldots \sum_{l_p=0}^{L-1} w_{op}(l_1, l_2, \ldots l_p) x(k-l_1) x(k-l_2) \ldots x(k-l_p)$$

- The *truncated* Volterra filter has P^{th} nonlinearity order and memory of length $L-1$

3.1 Volterra Filter Architecture

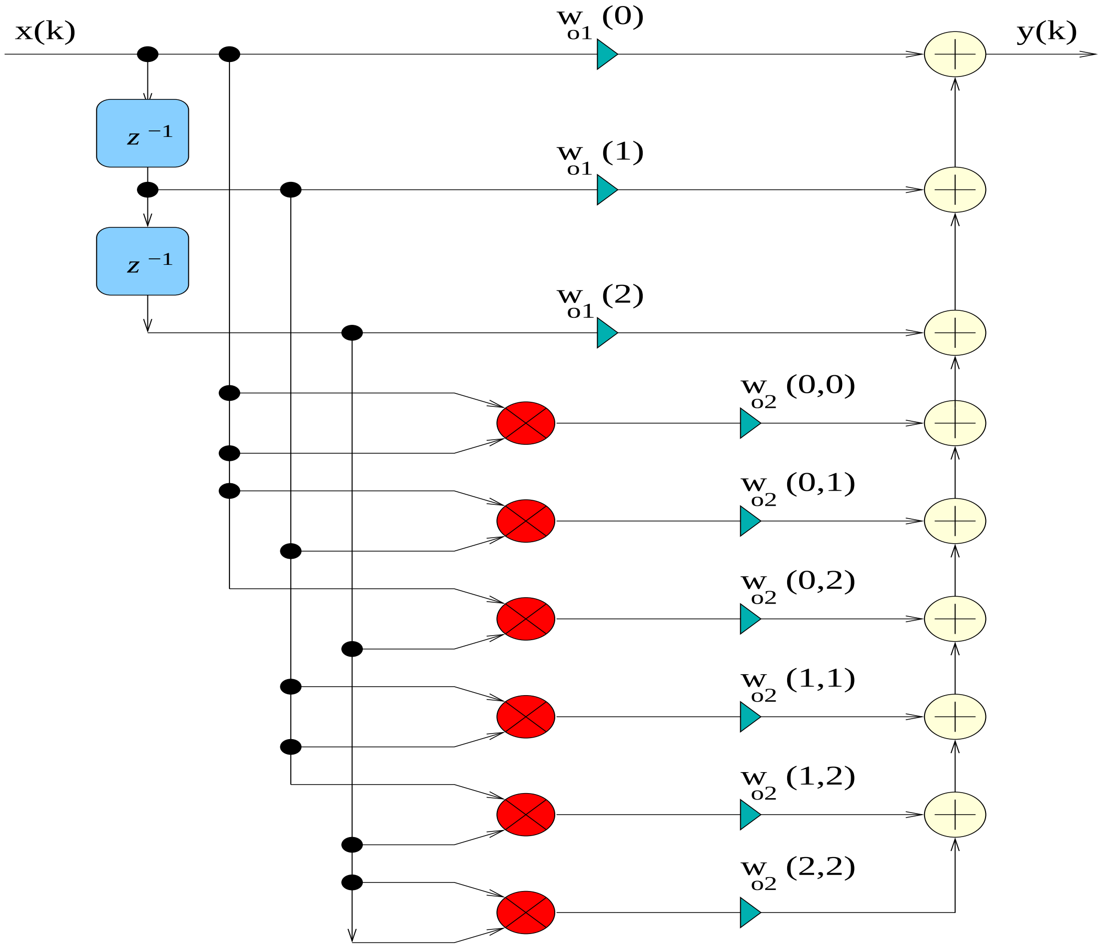

Figure 1: Truncated Volterra filter of order $P = 2$ and $L - 1 = 2$ delay elements

3.2 Volterra Series Expansion of Continuous Time Non-linear System

★ The continuous-time model:

$$
\begin{aligned}
y(t) \quad = \quad & \int_{-\infty}^{\infty} w_{o1}(\tau_1)x(t-\tau_1)d\tau_1 \\
& + \int\int_{-\infty}^{\infty} w_{o2}(\tau_1,\tau_2)x(t-\tau_1)x(t-\tau_2)d\tau_1 d\tau_2 \\
& + \int\int\int_{-\infty}^{\infty} w_{o3}(\tau_1,\tau_2,\tau_3)x(t-\tau_1)x(k-\tau_2)x(t-\tau_3)d\tau_1 d\tau_2 d\tau_3 \\
& + \int\int\ldots\int_{-\infty}^{\infty} w_{op}(\tau_1,\tau_2,\ldots\tau_p)x(t-\tau_1)x(t-\tau_2)x(k-\tau_p)d\tau_1 d\tau_2 \ldots d\tau_p + \\
& + \ldots
\end{aligned}
$$

4. Volterra Filters in Frequency Domain

- The Volterra model has also a frequency domain representation
- The p^{th} order kernel $w_{op}(\tau_1, \tau_2, \tau_3)$ is transformed using p-dimensional Fourier Transform
- (e.g. order $P = 3$):

$$H_3(\omega_1, \omega_2, \omega_3) = \int \int \int_{-\infty}^{\infty} w_{o3}(t_1, t_2, t_3) \exp[-j(\omega_1 t_1 + \omega_2 t_2 + \omega_3 t_3)] dt_1 dt_2 dt_3$$

- This representation allows us to obtain sinusoidal response, which is closely related to the harmonic distortion.

4. Volterra Filters in Frequency Domain

- Harmonic distortion and intermodulation products may be expressed in terms of the frequency response [3]:

$$HD_3 = \frac{A^2}{2} \cdot \frac{H_3(\omega, \omega, \omega)}{H_1(\omega)}$$

$$IM_3 = \frac{3A^2}{2} \cdot \frac{H_3(\omega, \omega, -\omega)}{H_1(\omega)}$$

5. Time Varying Systems

- The generalization of Volterra filters to the time-varying case is conceptually easy [3]
- The time-domain impulse response requires an additional time variable, so, $h_1(t, \tau)$ represents the system output at time t, if the impulse has been applied at time τ:

$$
\begin{aligned}
y(t) &= \int_{-\infty}^{\infty} h_1(t, \tau_1) x(\tau_1) d\tau_1 + \\
&+ \int\int_{-\infty}^{\infty} h_2(t, \tau_1, \tau_2) x(\tau_1) x(\tau_2) d\tau_1 d\tau_2 + \\
&+ \int\int\int_{-\infty}^{\infty} h_3(t, \tau_1, \tau_2, \tau_3) x(\tau_1) x(\tau_2) x(\tau_3) d\tau_1 d\tau_2 d\tau_3 + \\
&+ \ldots
\end{aligned}
$$

6. Nonlinear Adaptive Filtering

- A nonlinear filter cannot be described by an impulse response
- Nonlinear filters can be modeled by using polynomial models of non-linearity
- Volterra series expansion can model a large class of nonlinear filters and systems (semiconductors)
- Algorithms driven by Volterra series: LMS Volterra Filter, RLS Volterra Filter

6. Nonlinear Adaptive Filtering

- A nonlinear filter [1]

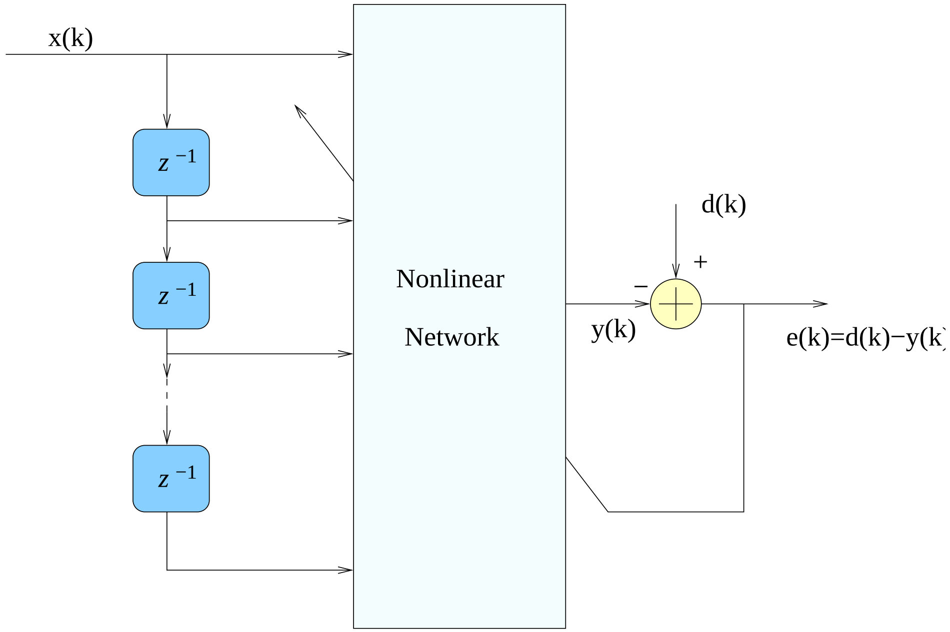

Figure 2: Adaptive nonlinear filter

6.1 The Vector Form of Truncated Volterra Series Expansion

- *Linear combination of nonlinear functions of the input signal* [2], the input-output relationship can be expressed easily in a vector form [4]:

$$y(k) = \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}$$

  where $y(k)$ is the output, and $\mathbf{x}(k)$ contains the nonlinear terms.

- For a Volterra filter with $P = 2$, the input is:

$$\mathbf{x}(k) = [1, \underbrace{x(k), \ldots, x(k-L+1)}_{L \text{ elements}} \,|\, \underbrace{x^2(k), x(k)x(k-1), \ldots x^2(k-L+1)}_{L(L+1)/2 \text{ elements}}]^{\mathrm{T}}$$

  and $\mathbf{w}$ contains all the kernel coefficients:

$$\mathbf{w} = [w_{o0}, w_{o1}(0), \ldots, w_{o1}(L-1) \,|\, w_{o2}(0,0), w_{o2}(1,0), \ldots, w_{o2}(L-1, L-1)]^{\mathrm{T}}$$

6.2 LMS Volterra Filter

- The objective function to be minimized is the Mean Square Error:

$$\mathrm{MSE} = \mathrm{E}[e^2(k)] = \mathrm{E}[\underbrace{(d(k) - y(k))^2}_{e^2(k)}]$$

- The instantaneous squared error $e^2(k) = [d(k) - \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k)]^2$ is minimized iteratively.
- The filter coefficients are adjusted according to the negative gradient direction:

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu \nabla_{\mathbf{w}} e^2(k)$$

- The gradient is:

$$\nabla_{\mathbf{w}} e^2(k) = \frac{\partial e^2(k)}{\partial \mathbf{w}} = 2e(k)\frac{\partial e(k)}{\partial \mathbf{w}} = -2e(k)\mathbf{x}(k)$$

6.2 LMS Volterra Filter

- It is wise to have different convergence factors for the different kernels (different nonlinearity order)
- For the particular case when the order is $P = 2$, we get:

$$
\begin{aligned}
w(l_1; k+1) &= w(l_1; k) + \mu_1 e(k) x(k - l_1) \\
w(l_1, l_2; k+1) &= w(l_1, l_2; k) + \mu_2 e(k) x(k - l_1) x(k - l_2)
\end{aligned}
$$

- *The convergence factors* are chosen according to:

$$
0 < \mu_1, \mu_2 < \frac{1}{\mathrm{trace}\{\mathbf{R}\}} < \frac{1}{\lambda_{\max}}
$$

where $\mathbf{R} = \mathrm{E}[\mathbf{x}(k)\mathbf{x}^{\mathrm{T}}(k)]$ and $\lambda_{\max} = \max_i\{\mathrm{eigval}_i[\mathbf{R}]\}$

- The convergence speed depends on the eigenvalue spreading

6.2 LMS Volterra Algorithm

- Initialization:

(1) $$\mathbf{x}(0) = \mathbf{w}(0) = [0, \ldots, 0]^{\mathrm{T}}$$

- For $k > 0$, the instantaneous error computation:

(2) $$e(k) = d(k) - \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k)$$

- The coefficient adjustment for a LMS Volterra filter of order P with $L-1$ delay elements:

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2e(k)[\mathrm{diag}\{\underbrace{\mu_1 \ldots, , \mu_1}_{L \text{ times}} | \underbrace{\mu_2, \ldots, \mu_2}_{L(L+1)/2 \text{ times}} | \ldots | \mu_P, \ldots, \mu_P\}]\mathbf{x}(k)$$

- In general LMS Volterra filter has a slow convergence speed, due to the eigenvalue spread (even with the whiteness assumption)

6.3 RLS Volterra Filter

- RLS algorithms are known to achieve fast convergence
- The objective function is different from the LMS case (exponential error weighting):

$$\mathcal{J}(k) = \sum_{i=0}^{k} \lambda^{k-i} e^2(i) = \sum_{i=0}^{k} \lambda^{k-i} [d(i) - \mathbf{x}^{\mathrm{T}}(i)\mathbf{w}(k)]^2$$

- The parameter λ controls *the memory span of the adaptive filter* $(0 < \lambda < 1)$
  By differentiating this function w.r.t. the filter coefficients $\mathbf{w}(k)$ and setting the derivative to zero:

$$\mathbf{w}(k) = \underbrace{\left[\sum_{i=0}^{k} \lambda^{k-i} \mathbf{x}(i)\mathbf{x}^{\mathrm{T}}(i) \right]^{-1}}_{\mathbf{R}_{\mathrm{D}}^{-1}(k)} \underbrace{\sum_{i=0}^{k} \lambda^{k-i} \mathbf{x}(i) d(i)}_{\mathbf{r}_{\mathrm{D}}(k)}$$

6.3 RLS Volterra Filter

- The optimal coefficients can be computed as:

$$\mathbf{w}(k) = \mathbf{R}_{\mathrm{D}}^{-1}(k)\mathbf{r}_{\mathrm{D}}(k)$$

- If we denote the deterministic correlation matrix of the input vector by:

$$\mathbf{R}_{\mathrm{D}}(k) = \sum_{i=0}^{k} \lambda^{k-i}\mathbf{x}(i)\mathbf{x}^{\mathrm{T}}(i) \qquad \text{and} \qquad \mathbf{R}_{\mathrm{D}}(k) = \lambda\mathbf{R}_{\mathrm{D}}(k-1) + \mathbf{x}(k)\mathbf{x}^{\mathrm{T}}(k)$$

and the deterministic cross-correlation vector between the input vector and the desired output:

$$\mathbf{r}_{\mathrm{D}}(k) = \sum_{i=0}^{k} \lambda^{k-i}\mathbf{x}(i)d(i) \qquad \text{and} \qquad \mathbf{r}_{\mathrm{D}}(k) = \lambda\mathbf{r}_{\mathrm{D}}(k-1) + \mathbf{x}(k)d(k)$$

6.3 RLS Volterra Algorithm

- Initialization: $\mathbf{x}(0) = \mathbf{w}(0) = [0, \ldots, 0]^{\mathrm{T}}$ and $\mathbf{R}_{\mathrm{D}}(-1) = \delta \mathbf{I}$
- For $k \geq 0$,

$$
\begin{aligned}
e'(k) &= d(k) - \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k-1) \\
\mathbf{g}(k) &= \frac{\lambda^{-1}\mathbf{R}_{\mathrm{D}}^{-1}(k-1)\mathbf{x}(k)}{1 + \lambda^{-1}\mathbf{x}^{\mathrm{T}}(k)\mathbf{R}_{\mathrm{D}}^{-1}(k-1)\mathbf{x}(k)} \\
\mathbf{w}(k) &= \mathbf{w}(k-1) + \mu \mathbf{g}(k) e'(k) \\
\mathbf{R}_{\mathrm{D}}^{-1}(k) &= \lambda^{-1}\mathbf{R}_{\mathrm{D}}^{-1}(k-1) - \lambda^{-1}\mathbf{g}(k)\mathbf{x}^{\mathrm{T}}(k)\mathbf{R}_{\mathrm{D}}^{-1}(k-1)
\end{aligned}
$$

- If necessary compute:

$$
\begin{aligned}
\mathbf{y}(k) &= \mathbf{w}^{\mathrm{T}}(k)\mathbf{x}(k) \\
e(k) &= d(k) - \mathbf{x}^{\mathrm{T}}(k)\mathbf{w}(k)
\end{aligned}
$$

Summary and Conclusions

- Volterra series can model a large class of nonlinear systems
- The expansion is a *linear combination of nonlinear functions of the input signal*
- Both LMS and RLS are used in practice to identify unknown time-invariant systems
- The LMS Algorithm
  - simple
  - computationally efficient (e.g.*Sign LMS Algorithm*)
  - suffers from low convergence speed
- The RLS Algorithm
  - faster than LMS
  - theoretically it achieves the optimal solution (Wiener solution)
  - more complex (matrix inversion)

Bibliography

References

[1] Paulo S. R. Diniz - "Adaptive Filtering Algorithms and Practical Implementation", 2nd Ed., *Kluwer Academic Publishers, 2002*

[2] V. John Mathews - “Adaptive Polynomial Filters” *IEEE Signal Processing Magazine*, Volume:8 Issue:3, July 1991, pp:10-26

[3] Wei Yu, Subhajit Sen, and Bosco Leung, - “Time Varying Volterra Series and Its Applications to the Distortion Analysis of a Sampling Mixer”, *Circuits and Systems, 1997. Proceedings of the 40th Midwest Symposium on* , Volume:1 , 3-6 Aug. 1997, pp:245-248 vol.1

[4] Ian J. Morrison and Peter J.W. Rayner - “The Application of Volterra Series to Signal Estimation”, *Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference on*, 14-17 April 1991, pp: 1481-1484 vol.2