VOLTERRA FILTER IDENTIFICATION USING PENALIZED LEAST SQUARES

Robert D. Nowak

Department of Electrical & Computer Engineering
Rice University, Houston, TX 77005 USA

ABSTRACT

Volterra filters have been applied to many nonlinear system identification problems. However, obtaining good filter estimates from short and/or noisy data records is a difficult task. We propose a penalized least squares estimation algorithm and derive appropriate penalizing functionals for Volterra filters. An example demonstrates that penalized least squares estimation can provide much more accurate filter estimates than ordinary least squares estimation.

1. INTRODUCTION

Volterra filters have been applied to many nonlinear system identification problems (see [2, 3] for a summary of applications). It is well known that polynomial regression models, such as the Volterra filter, often suffer from severe ill-conditioning [5]. Consequently, least squares Volterra filter estimates obtained from noisy data are often very poor. In this paper, we describe a estimation algorithm based on the method of penalized least squares (PLS) [1, 6]. PLS is a well known method for regularizing the LS estimator and is shown to significantly improve Volterra filter estimates. The main contribution of this paper is the design of appropriate penalizing functionals for the Volterra filter estimation problem.

2. VOLTERRA FILTER IDENTIFICATION

The output of a pth order Volterra filter with memory m in response to a real-valued input $\{x(k)\}_{k\in\mathbb{Z}}$ is given by

$$\begin{aligned} v(k) &= \sum_{j=1}^{p} \sum_{k_1,\dots,k_j=1}^{m} h_j(k_1,\dots,k_j) \\ &\quad \times x(k-k_1)\cdots x(k-k_j), \end{aligned} \tag{1}$$

Supported by the National Science Foundation, grant no. MIP–9457438, and the Office of Naval Research, grant no. N00014–95–1–0849.

where h_j, referred to as an jth order Volterra kernel, is deterministic and is real-valued[1]. Note that certain products are unnecessarily repeated in (1), and hence we may assume that the kernels are symmetric. Let $\mathbf{x}(k)$ be a column vector whose elements are the unique products in (1), and let $\boldsymbol{\theta}_h$ be the column vector of the unique kernel parameters (accounting for symmetries) so that (1) is rewritten as

$$v(k) = \mathbf{x}(k)^T\boldsymbol{\theta}_h. \tag{2}$$

One of the most important aspects of the Volterra filter is that the kernel parameters are linearly related to the output. Therefore, given the filter input and output, identifying the kernel parameters is a linear estimation problem. In a typical identification experiment, we have a finite number of input and output measurements. In most cases, it is assumed that the output measurements are contaminated by an additive i.i.d. observation noise. This noise may represent errors due to mismodelling, sampling, quantization, or sensor noise. The observed output is given by $y(k) = v(k)+\eta(k)$, where $\eta(k)$ is the observation noise. Given a finite set of input $\{x(k) : k = -m,\dots,n-1\}$ and noisy output $\{y(k) : k = 0,\dots,n-1\}$ measurements, estimates of the Volterra kernels can be obtained using least squares (LS). Let $\mathbf{y} = [y(0),\dots,y(n-1)]^T$, $\boldsymbol{\eta} = [\eta(0),\dots,\eta(n-1)]^T$, and $\mathbf{X} = [\mathbf{x}(0),\dots,\mathbf{x}(n-1)]^T$; then $\mathbf{y} = \mathbf{X}\boldsymbol{\theta}_h + \boldsymbol{\eta}$. Assuming that the matrix $\mathbf{X}^T\mathbf{X}$ is invertible, the unique least squares estimate of $\boldsymbol{\theta}_h$ is

$$\widehat{\boldsymbol{\theta}}_h = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}, \tag{3}$$

and $\widehat{\boldsymbol{\theta}}_h$ minimizes the residual sum of squared errors

$$\frac{1}{n}\|\mathbf{y}-\mathbf{X}\boldsymbol{\theta}_h\|_2^2. \tag{4}$$

[1] It is not necessary that the input to the Volterra filter be a time sequence. For example, the m input samples may be pixels in an image or measurements from an m-dimensional sensor array.

3. PENALIZED LEAST SQUARES

In general, Volterra filter identification is "ill-posed". A standard approach to ill-posed inverse problems is known as the *method of regularization*. A simple regularization procedure for LS problems is the method of *penalized least squares* (PLS) [1, 6].

In PLS, the square error criterion (4) is augmented with a penalizing functional to form a new criterion function given by

$$\frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}_h\|_2^2 + \lambda\, J(\boldsymbol{\theta}_h), \qquad \lambda > 0. \tag{5}$$

The penalizing functional J is chosen to reflect partial prior information that may be available regarding the unknown $\boldsymbol{\theta}_h$. J is non-negative and is chosen to weight undesirable or unlikely solutions more heavily than desirable or likely solutions.

The parameter λ in (5), called the regularization or smoothing parameter, controls the trade-off between the fidelity to the data, measured by the squared error $\|\mathbf{y} - \mathbf{X}^T\boldsymbol{\theta}_h\|_2^2$, and the penalty $J(\boldsymbol{\theta}_h)$. Regularization reduces the estimator variance at the expense of possibly introducing a bias. There are many automatic procedures for choosing a good regularization parameter λ. Theoretical and empirical evidence shows that such methods often provide very good results [1, 6].

The critical issue in PLS is designing an appropriate penalizing functional J. As mentioned above, this choice should reflect prior information that may be known about the problem at hand. If J is chosen wisely, then the bias of the penalized estimator will be negligible. In many applications, a reasonable penalizing functional is expressed as a quadratic form; for a general penalized least squares problem $J(\boldsymbol{\theta}) = \boldsymbol{\theta}^T\mathbf{R}\boldsymbol{\theta}$, where $\mathbf{R}$ is a positive semidefinite matrix. The next section examines the design of $\mathbf{R}$ for the Volterra filter identification problem.

4. PENALIZING FUNCTIONALS FOR VOLTERRA FILTER IDENTIFICATION

The derivation of penalizing functionals is accomplished in three steps. First, we introduce a tensor product representation of the Volterra filter. With this representation, the relationship between kernel penalties and filter response is easily described. This relationship allows us to construct meaningful and useful kernel penalizing functionals. The third step transforms the kernel penalizing functionals into penalizing functionals for the underlying parameter vector $\boldsymbol{\theta}_h$. To simplify the presentation, throughout this section we will concentrate on the pth order homogeneous Volterra filter (*i.e.,* the Volterra filter involving only p-fold products of the input). Extensions to the general Volterra filter are straightforward.

4.1. Tensor Product Representation

In the pth order homogeneous case, equation (2) relates the output $v(k)$ to the unique parameters of the pth order kernel h_p. An alternative vector representation using the tensor (Kronecker) product [4] is useful for interpreting the relationship between the kernel and the input. We use the following notation throughout the paper. For any matrix or vector $\mathbf{A}$, let $\mathbf{A}^{(p)}$ denote the p-fold tensor (Kronecker) product of $\mathbf{A}$ with itself.

The tensor product representation of the Volterra filter is derived as follows. First, let

$$\mathbf{d}(k) = [x(k), \ldots, x(k-m+1)]^T,$$

a vector of the input samples needed to compute the Volterra filter output $v(k)$. Note that the tensor product $\mathbf{d}^{(p)}(k)$ contains all p-fold input products involved in the Volterra filter. The Volterra filter may be written as

$$v(k) = \mathbf{h}_p^T \mathbf{d}^{(p)}(k), \tag{6}$$

where $\mathbf{h}_p$ is an $m^p \times 1$ vector containing all elements of the Volterra kernel h_p. The difference between (2) and (6) is that the symmetries inherent in the Volterra filter are not accounted for in (6). The advantage of (6) is that the output is related directly to the kernel rather than the parameter vector $\boldsymbol{\theta}_h$. This is crucial to the derivation of appropriate penalty functionals.

4.2. Kernel Penalizing Functionals

The goal of the penalizing functional is to penalize solutions that are unlikely or undesirable. The input response of the Volterra filter characterizes its performance. Therefore, a useful penalizing functional for the Volterra filter is one that penalizes response behavior that is unlikely or undesirable. Specifically, assume that prior information suggests that it is unlikely or undesirable for the unknown Volterra filter to produce a strong response to a specific subspace of the input space. This subspace is called the *penalized subspace.*

Penalizing functionals for the Volterra filter are derived from the penalized subspace as follows. Let $\mathbf{P}$ denote the $m \times m$ orthogonal projector corresponding to the penalized subspace and for any matrix or vector $\mathbf{A}$, let $\mathbf{A}^{(p)}$ denote the p-fold tensor (Kronecker) product of $\mathbf{A}$ with itself. Using some simple tensor product identities [4] the Volterra filter $v(k) = \mathbf{h}_p^T\mathbf{d}^{(p)}(k)$ may be decomposed as follows:

$$v(k) = (\mathbf{P}^{(p)}\mathbf{h}_p)^T\mathbf{d}^{(p)}(k) + ([\mathbf{I}^{(p)} - \mathbf{P}^{(p)}]\mathbf{h}_p)^T\mathbf{d}^{(p)}(k), \tag{7}$$

where $\mathbf{I}$ is the $m \times m$ identity matrix. It is easily shown that

$$(\mathbf{P}^{(p)}\mathbf{h}_p)^T \mathbf{d}^{(p)}(k) = \mathbf{h}_p^T (\mathbf{P}\mathbf{d}(k))^{(p)}.$$

Hence, $(\mathbf{P}^{(p)}\mathbf{h}_p)^T \mathbf{d}^{(p)}(k)$ is the filter response to the input component in the penalized subspace. $\mathbf{P}^{(p)}\mathbf{h}_p$ is called the *penalized kernel* and is simply the projection of the original kernel $\mathbf{h}_p$ onto the p-fold tensor product of the penalized subspace.

Now consider a penalizing functional of the form

$$J(\mathbf{h}_p) = \mathbf{h}_p^T \mathbf{P}^{(p)} \mathbf{h}_p. \tag{8}$$

The decomposition (7) shows that (8) weights the penalized kernel only. This penalizing functional is easily generalized to

$$J(\mathbf{h}_p) = \mathbf{h}_p^T \mathbf{R}^{(p)} \mathbf{h}_p, \tag{9}$$

where $\mathbf{R}$ is a symmetric, positive semidefinite matrix whose dominant eigenvectors (associated with the large eigenvalues) span the penalized subspace. The generalization (9) allows one to weight the relative amount of penalization applied to the Volterra kernel. A concrete example of Volterra kernel penalizing functional is given in the example of Section 5. The choice of penalizing functional is closely related to the problem of Volterra filter approximation [4]. Many of the results in [4] are directly applicable to the choice and design of appropriate penalizing functionals.

The tensor product representation suggests a useful way of constructing penalizing functionals for the Volterra kernels. However, the PLS problem is formulated with the kernel parameter vector $\boldsymbol{\theta}_h$ rather than the kernel $\mathbf{h}_p$ itself. Therefore, the Volterra kernel penalizing matrix $\mathbf{R}^{(p)}$ must be modified to apply to the parameter vector $\boldsymbol{\theta}_h$.

4.3. Penalizing Functionals for $\boldsymbol{\theta}_h$

To obtain a penalizing functional for the parameter vector we need to construct transformations relating $\mathbf{h}_p$ and $\boldsymbol{\theta}_h$. Linear transformations $\mathbf{T}_p$ and $\mathbf{U}_p$ satisfying $\boldsymbol{\theta}_h = \mathbf{T}_p\mathbf{h}_p$ and $\mathbf{h}_p = \mathbf{U}_p\boldsymbol{\theta}_h$ are easily determined according the symmetries of the kernel $\mathbf{h}_p$. Each p-tuple $(i_1, \ldots, i_p)$ corresponds to an element $h_p(i_1, \ldots, i_p)$ of the kernel and for every permutation $\sigma(1), \ldots, \sigma(p)$ of $1, \ldots, p$ we have

$$h_p(i_1, \ldots, i_p) = h_p(i_{\sigma(1)}, \ldots, i_{\sigma(p)}).$$

Hence, with each p-tuple we identify its unique generating p-tuple $(i_1, \ldots, i_p)$, where $i_1 \leq i_2 \leq \cdots \leq i_p$. Let $\{g_k\}_{k=1}^L$ be the set of unique generating p-tuples, where $L = \binom{p+m-1}{p}$, the binomial coefficient.

The matrix $\mathbf{T}_p$ is constructed as follows. Let n_k be the number of distinct permutations of g_k. For example, if $g_k = (1,1,2,5)$, then $n_k = \frac{4!}{2!1!1!} = 12$. Let $j_{1,k}, \ldots, j_{n_k,k}$ denote the positions in $\mathbf{h}_p$ of $h_p(g_k) = h_p(i_{1,k}, \ldots, i_{p,k})$ and the identical kernel elements associated with the $n_k - 1$ permutations of g_k. Notice that the positions $j_{1,k}, \ldots, j_{n_k,k}$ are dictated by the tensor product formation of $\mathbf{d}^{(p)}(k)$. Now let the elements in kth row of $\mathbf{T}_p$ take the value 1 at the positions $j_{1,k}, \ldots, j_{n_k,k}$ and 0 elsewhere. It is easily verified that $\boldsymbol{\theta}_h = \mathbf{T}_p\mathbf{h}_p$.

The transformation $\mathbf{U}_p$ is easily constructed from $\mathbf{T}_p$. First, divide each row of $\mathbf{T}_p$ by its sum (= number of 1's in the row) to obtain a matrix $\widetilde{\mathbf{T}}_p$. Then $\mathbf{U}_p = \widetilde{\mathbf{T}}_p^T$.

To illustrate the construction, consider the simple case of a quadratic kernel ($p = 2$) with memory $m = 2$. In this case, $\mathbf{h}_2 = [h_2(1,1),\, h_2(1,2),\, h_2(2,1),\, h_2(2,2)]^T$,

$$\mathbf{T}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

and $\boldsymbol{\theta}_h = \mathbf{T}_2\mathbf{h}_2 = [h_2(1,1),\, 2\,h_2(1,2),\, h_2(2,2)]^T$. The matrix $\mathbf{U}_2$ is given by

$$\mathbf{U}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Continuing, recall that the kernel penalizing functional takes the form

$$J(\mathbf{h}_p) = \mathbf{h}_p^T \mathbf{R}^{(p)} \mathbf{h}_p, \tag{10}$$

where $\mathbf{R}^{(p)}$ is a positive semidefinite matrix as defined in (9). Replacing $\mathbf{h}_p$ by $\mathbf{U}_p\boldsymbol{\theta}_h$ produces

$$\begin{aligned} J(\boldsymbol{\theta}_h) &= (\mathbf{U}_p\boldsymbol{\theta}_h)^T \mathbf{R}^{(p)} \mathbf{U}_p \boldsymbol{\theta}_h, \\ &= \boldsymbol{\theta}_h^T (\mathbf{U}_p^T \mathbf{R}^{(p)} \mathbf{U}_p) \boldsymbol{\theta}_h. \end{aligned} \tag{11}$$

Equation (11) provides the form of the parameter vector penalizing functional. Given an appropriate kernel penalizing matrix $\mathbf{R}^{(p)}$, a parameter vector penalizing matrix is obtained by the transformation $\mathbf{U}_p^T \mathbf{R}^{(p)} \mathbf{U}_p$. Hence, the PLS Volterra filter criterion function is

$$\frac{1}{n}\,\|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}_h\|_2^2 + \lambda\, \boldsymbol{\theta}_h^T (\mathbf{U}_p^T \mathbf{R}^{(p)} \mathbf{U}_p) \boldsymbol{\theta}_h, \quad \lambda > 0. \tag{12}$$

Assuming that $(\mathbf{X}^T\mathbf{X} + n\,\lambda\,\mathbf{U}_p^T\mathbf{R}^{(p)}\mathbf{U}_p)$ is invertible, it follows from Theorem 2 in [1] that the unique minimizer of (12) is given by

$$\widehat{\boldsymbol{\theta}}_h(\lambda) = (\mathbf{X}^T\mathbf{X} + n\,\lambda\,\mathbf{U}_p^T\mathbf{R}^{(p)}\mathbf{U}_p)^{-1}\mathbf{X}^T\mathbf{y}. \tag{13}$$

Remark: Note that invertibility of $\mathbf{X}^T\mathbf{X}$ guarantees that $(\mathbf{X}^T\mathbf{X} + n\,\lambda\,\mathbf{U}_p^T\mathbf{R}^{(p)}\mathbf{U}_p)$ is invertible. However, $(\mathbf{X}^T\mathbf{X}+n\,\lambda\,\mathbf{U}_p^T\mathbf{R}^{(p)}\mathbf{U}_p)$ can be invertible even if $\mathbf{X}^T\mathbf{X}$ is singular. This observation is very important for identification from short data records, a situation in which $\mathbf{X}^T\mathbf{X}$ often fails to be invertible [5].

5. NUMERICAL EXAMPLE

In this section we illustrate the PLS identification of a quadratic Volterra filter. The quadratic kernel of the unknown filter is depicted in Fig. 1 (a). Visual inspection shows that the true kernel is fairly smooth and hence a roughness penalty is appropriate. In practice the true kernel is unknown. However, we may deduce the underlying smoothness of the Volterra kernel in a number of ways. For example, if the frequency content of the filter output is sufficiently lowpass, then a roughness penalty is in order. Further characterization is possible by applying special test inputs to "probe" the unknown system prior to complete identification.

In our example, we choose a kernel penalizing matrix of the form $\mathbf{R}^{(2)} = (\mathbf{I}-\mathbf{S})^{(2)}$ where $\mathbf{S}$ is the projection onto a low dimensional subspace spanned by the discrete prolate spheroidal sequences (DPSS) [7] corresponding to the cutoff frequency f_0. The DPSS are the optimal sequences of length m that concentrate the largest fraction of the total energy in the lowpass band $[-f_0, f_0) \subset [-1/2, 1/2)$. The parameter vector penalizing matrix is obtained via the transformation in (11).

To demonstrate the performance of PLS compared to LS we simulate a system identification experiment. The input $\{x(k)\}$ is generated i.i.d. $N(0,1)$. The cutoff frequency for the smoothing matrix is $f_0 = \frac{1}{8}$ and $\operatorname{rank}\mathbf{S} = 10$. The observation noise $\{\eta(k)\}$ is i.i.d. $N(0, 0.1)$. For the simulation, we use $n = 2000$ input and output observations. Fig. 1 (b) shows the LS estimate. The squared error of the LS estimate is 0.0421. The smoothing parameter for the PLS estimate is $\lambda = 3.98$; λ is determined using the method of generalized cross validation [1]. The PLS estimate is depicted in Fig. 1 (c). The squared error of the PLS estimate is 0.0063, nearly an order of magnitude smaller than the squared error of the LS estimate.

6. CONCLUSIONS

This paper demonstrates the utility of PLS for obtaining good Volterra kernel estimates from noisy data. Prior information can be used to derive appropriate penalizing functionals for the problem at hand. Generalized cross validation automatically adjusts the amount of penalization and therefore the estimates are not unduly constrained by the prior information.

REFERENCES

[1] M. Von Golitschek and L. L. Schumaker, "Data fitting by penalized least squares," in *Algorithms for approximation, II*, edited by J.C. Mason and M.G. Cox, New York, pp. 210-227, Chapman and Hall, 1990.

[2] P. Z. Marmarelis and V. Z. Marmarelis, *Analysis of Physiological Systems. The White Noise Approach*, Plenum Press, New York, 1978.

[3] S. W. Nam and E. J. Powers, "Application of higher order spectral analysis to cubically nonlinear system identification," *IEEE Tran. Signal Processing*, vol. 42, no. 7, pp. 2124-2135, July 1994.

[4] R. D. Nowak and B. D. Van Veen, "Tensor Product Basis Approximations for Volterra Filters," *IEEE Transactions on Signal Processing*, vol. 44, no. 1, January 1996.

[5] R. D. Nowak and B. D. Van Veen, "Random and pseudorandom inputs for Volterra filter identification," *IEEE Tran. Signal Processing*, vol. 42, no. 8, pp. 2124-2135, August 1994.

[6] F. O'Sullivan, "A statistical perspective on ill-posed inverse problems," *Statistical Science*, vol. 1, no. 4, pp. 502-527, 1986.

[7] D. Slepian, "Prolate spheroidal wave functions, Fourier analysis, and uncertainty-V: The discrete case," *Bell Syst. Tech. J.*, vol. 40, pp. 1371-1429, 1978.

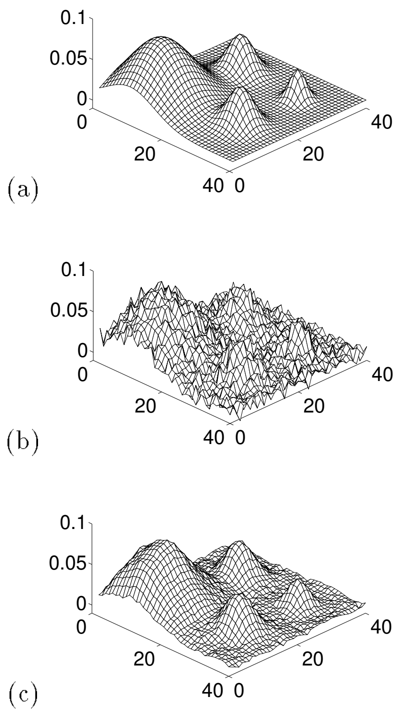

Figure 1: *Quadratic system identification: (a) Unknown kernel (b) LS estimate (c) PLS estimate*