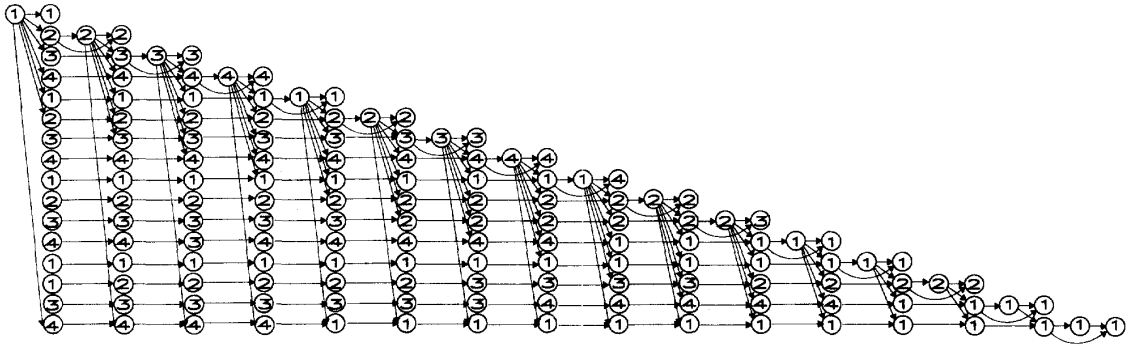

Fig. 5. Mapping of full QR using the Min-Max + Idle cost function.

matrix using a four processor, bidirectional ring. The communication and computation parameters were set to model the iPSC/860 multiprocessor. Deviations from column-wrapped input and output were penalized as in Section III-D. The mapping that resulted is shown in Fig. 5. Experiments were then conducted on the iPSC/860 to measure the relative performance of this mapping compared with the column-wrap mapping and several random mappings. We found that the column-wrap mapping was 16.75% slower that the mapping of Fig. 5. On average, random mappings were 129.5% slower than our mapping.

VI. Conclusion

We have presented a method for mapping numerical algorithms onto multiprocessors. Through case studies of two important DSP problems, we showed that the cost function is the critical element in achieving good mappings. Our experiments on the iPSC/860 hypercube indicate that we have succeeded in our goal of defining cost functions that are both accurate and tractable. Our experiments with the QR decompostion on the iPSC/860 confirm that our mappings are highly concurrent. The approach holds great promise for bringing parallel processing to bear on the problems in signal processing as it frees the user from the need to have an in-depth knowledge of parallel programming.

References

[1] A. W. Bojanczyk and J. Choi, "Recursive least squares problems in distributed memory multiprocessors," to be published in *J. Parallel Distr. Comput.*

[2] C. S. Henkel, M. T. Heath, and R. J. Plemmons, "Cholesky downdating on a hypercube," in *Proc. 3rd Conf. Hypercube Concurrent Comput. Applications*, 1988, pp. 1592–1598.

[3] C.-C. Shen and W.-H. Tsai, "A graph matching approach to optimal task assignment in distributed computing systems using a minimax criterion," *IEEE Trans. Comput.*, vol. C-34, pp. 197–203, 1985.

[4] R. A. Widlicka, "Mapping algorithms onto hypercubes using simulated annealing," in *Proc. 4th Conf. Hypercube Concurrent Comput.*, 1989, pp. 71–74.

[5] F. Ercal, J. Ramanujam, and P. Sadayappan, "Task allocation onto a hypercube by recursive mincut bipartitioning," in *Proc. 3rd Conf. Hypercube Concurrent Comput.*, 1988, pp. 210–221.

[6] K. Konstantinides, R. Kaneshiro, and J. Tani, "Task allocation and scheduling models for mulitprocessor digital signal processing," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, pp. 2151–2161, 1990.

[7] F. Andre, J.-L. Pazat, and T. Priol, "Experiments with mapping algorithms on a hypercube," in *Proc. 4th Conf. Hypercube Concurrent Comput.*, 1989, pp. 39–46.

[8] C. C. Pettey and M. R. Leuze, "Parallel placement of parallel processes," in *Proc. 3rd Conf. Hypercube Concurrent Comput.*, 1988, pp. 232–238.

[9] H. Dijkstram *et al.*, "A general-purpose video signal processor: Architecture and programming," in *Proc. Int. Conf. Comput. Design*, 1989, pp. 74–77.

[10] C. M. Rader and A. O. Steinhardt, "Hyperbolic Householder transforms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 1589–1602, 1986.

[11] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore: Johns Hopkins University Press, 1989, 2nd ed.

[12] P. Hoang and J. Rabaey, "Scheduling of DSP programs onto multiprocessors for maximum throughput," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, pp. 2225–2235, 1990.

[13] S. Kirkpatrick, C. Gelatt, Jr., and M. Vecchi, "Optimization by simulated annealing," *Sci.*, vol. 220, pp. 671–680, 1983.

[14] D. Rabideau and A. Steinhardt, "Simulated annealing for mapping DSP algorithms onto multiprocessors," in *Proc. 27th Asilomar Conf. Signals, Syst., Comput.*, 1993.

Efficient Computational Schemes for the Orthogonal Least Squares Algorithm

E. S. Chng, S. Chen, and B. Mulgrew

***Abstract*— The orthogonal least squares (OLS) algorithm is an efficient implementation of the forward selection method for subset model selection. The ability to find good subset parameters with only a linearly increasing computational requirement makes this method attractive for practical implementations. In this correspondence, we examine the computational complexity of the algorithm and present a preprocessing method for reducing the computational requirement.**

I. Introduction

Nonlinear predictors generated using radial basis functions (RBF) [2], [4], fuzzy basis functions [5] or Volterra expansions [6] normally result in the formation of very large initial models that have the

Manuscript received June 21, 1993; revised July 8, 1994. The associate editor coordinating the review of this paper and approving it for publication was Prof. J. N. Hwang.

E. S. Chng and B. Mulgrew are with the Department of Electrical Engineering, The University of Edinburgh, Edinburgh EH9 3JL, Scotland, UK.

S. Chen is with the Department of Electrical and Electronics Engineering, The University of Portsmouth, Portsmouth PO1 3DJ, England, UK.

IEEE Log Number 9406917.

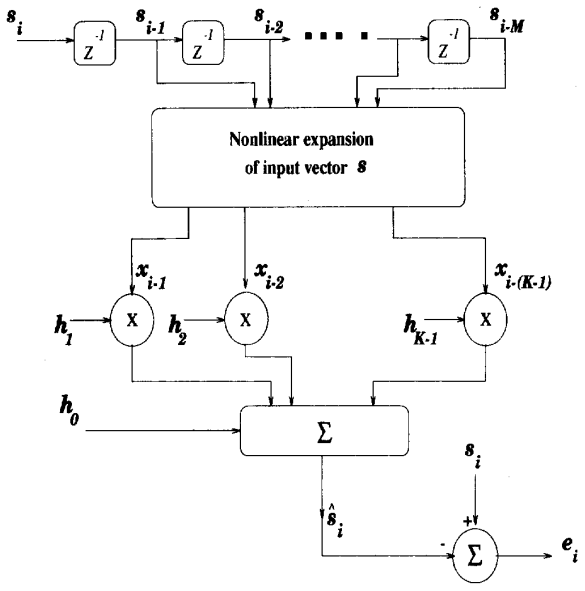

Fig. 1. Nonlinear predictor of order K.

linear-in-parameter characteristic (Fig. 1). Such large initial models can normally be reduced to a much smaller parsimonious model without significant degradation in prediction performance if the subset model's parameters are chosen carefully.

To find the optimum R-parameter subset model from an original K-parameter model, it is required to calculate the performance of all the possible R-parameter subset models from the original K-parameter system and choose the best one. This requires prohibitively large amount of computation and is thus not practical.

One applicable method of subset model selection for models with the linear-in-parameter characteristic is the forward-selection search [3]. This method, however, has been criticized for not guaranteeing to achieve the optimum solution. Although the criticism is valid, subset models found using the forward-selection search are generally good enough for practical applications. Examples can be found in the papers describing the OLS algorithm [1], [2] and Korenbergh's fast orthogonal search [7]; these two methods are derivatives of the forward-selection technique.

II. OLS Algorithm

Let us represent these nonlinear predictors that have the linear in parameter structure as a linear regression model:

$$\mathbf{y} = \mathbf{X}\mathbf{h} + \mathbf{e} \tag{1}$$

where $\mathbf{y}$ is the desired signal vector, $\mathbf{X}$ is the information matrix of size $N \times K$, $\mathbf{h}$ is the parameter vector of the model, and $\mathbf{e}$ the error vector of approximating $\mathbf{y}$ by $\mathbf{X}\mathbf{h}$. The column vectors $\mathbf{y}$ and $\mathbf{e}$ contain N elements, that is, there are N data samples and N values of error.

The original $\mathbf{X}$ matrix has K columns. To create a parsimonious model which has R parameters, we are actually trying to pick R columns from the input matrix $\mathbf{X}$ to form a subset input matrix $\mathbf{X}_S$. The OLS algorithm selects columns from the input matrix sequentially. At each selection, all the unused columns are studied to determine how each column will contribute to fit the desired vector $\mathbf{y}$ with the current subset $\mathbf{X}_S$. The column that provides the best combination with $\mathbf{X}_S$ to model $\mathbf{y}$ will be picked to form the new $\mathbf{X}_S$. The above procedure is repeated until the number of columns in $\mathbf{X}_S$ equals to R. The selection procedure is made very efficient by employing orthogonalization schemes such as the Gram-Schmidt or the Householder transformation [8]. The details of the algorithm can be found in Chen *et al.* [1], [2].

III. Computation Reduction of OLS Method

The computational requirement in applying the OLS algorithm to find subset models from an initial information matrix $\mathbf{X}$ is proportional to the size of $\mathbf{X}$. In the situation when $N \gg K$, where N and K are the numbers of rows and columns in $\mathbf{X}$ respectively, it may be possible to reduce computation requirement of the OLS by first introducing an invariant transformation on the matrix $\mathbf{X}$ and then applying the OLS on the transformed data.

This is accomplished by premultiplying (1) by an orthonormal matrix which spans the column space of $\mathbf{X}$ [8] to transform the $N \times K$ matrix $\mathbf{X}$ and the $N \times 1$ vector $\mathbf{y}$ into a $K \times K$ matrix $\tilde{\mathbf{X}}$ and a $K \times 1$ vector $\tilde{\mathbf{y}}$. This may be thought of as a preprocessing. The OLS algorithm is then employed to select subset model based on $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{y}}$.

A. Reduced-OLS Gram-Schmidt Approach

We first examine the classical Gram-Schmidt (GS) procedure [8] for generating the orthonormal matrix used for the invariant transformation. The information matrix $\mathbf{X}$ can be decomposed into the product of an $N \times K$ matrix $\mathbf{Q}$ satisfying $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$ and a $K \times K$ upper triangle matrix $\mathbf{B}$, where $\mathbf{I}$ is the identity matrix of appropriate dimension. That is

$$\mathbf{X} = \mathbf{Q}\mathbf{B}. \tag{2}$$

Premultiplying both sides of (1) by $\mathbf{Q}^T$ yields

$$\mathbf{Q}^T\mathbf{y} = \mathbf{B}\mathbf{h} + \mathbf{Q}^T\mathbf{e}. \tag{3}$$

If we introduce $\tilde{\mathbf{y}} = \mathbf{Q}^T\mathbf{y}$, $\tilde{\mathbf{X}} = \mathbf{B}$, and $\tilde{\mathbf{e}} = \mathbf{Q}^T\mathbf{e}$, we can rewrite (3) as

$$\tilde{\mathbf{y}} = \tilde{\mathbf{X}}\mathbf{h} + \tilde{\mathbf{e}}. \tag{4}$$

$\tilde{\mathbf{y}}$ and $\tilde{\mathbf{e}}$ are $K \times 1$ vectors and $\tilde{\mathbf{X}}$ is a $K \times K$ matrix. We can then apply the OLS algorithm to perform subset selection based on $\tilde{\mathbf{y}}$ and $\tilde{\mathbf{X}}$. We call this method the reduced-OLS GS approach.

An R-term subset model found using the reduced-OLS GS approach is identical to that of applying the OLS on the original data. This is because the transformed data $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{y}}$ are created by performing a unitary transformation [8] on $\mathbf{X}$ and $\mathbf{y}$. As such a transformation preserves the length of each (column) vector and the angle between two vectors, we have not lost or created any new information when we transform (1) into (4).

The amount of computation required to apply the invariant transformation by this method requires approximately $N \times K^2$ multiplications [8]. If saving in computation by using $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{y}}$ for subset selection offsets the additional computation of preprocessing, this reduced OLS approach is justified. Computational complexity of this reduced OLS algorithm for subset model selection has been analyzed, and we illustrate the results using an example showing the number of multiplications needed for subset selection working on a 500×84 information matrix (Fig. 2). The horizontal axis of Fig. 2 shows the size of the selected subset model, and the vertical axis shows the number of multiplications performed by the OLS or the reduced-OLS GS algorithm to find the required subset model.

The following equation can be used to calculate the number of multiplications performed by the OLS to select a subset model of R

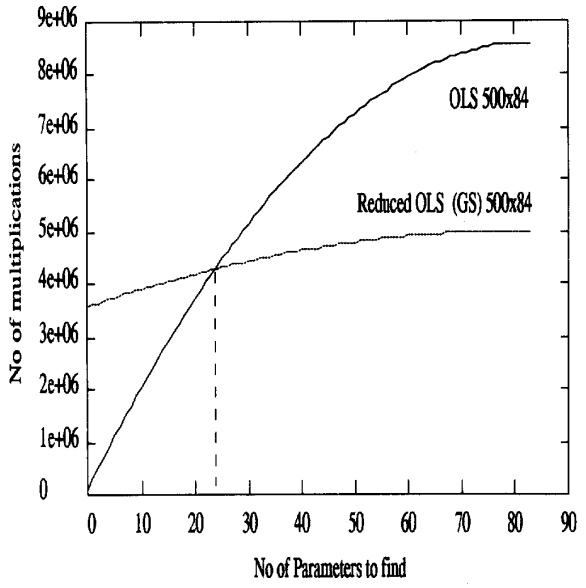

Fig. 2. Computation requirement for $\mathbf{X}$ matrix of size 500 × 84.

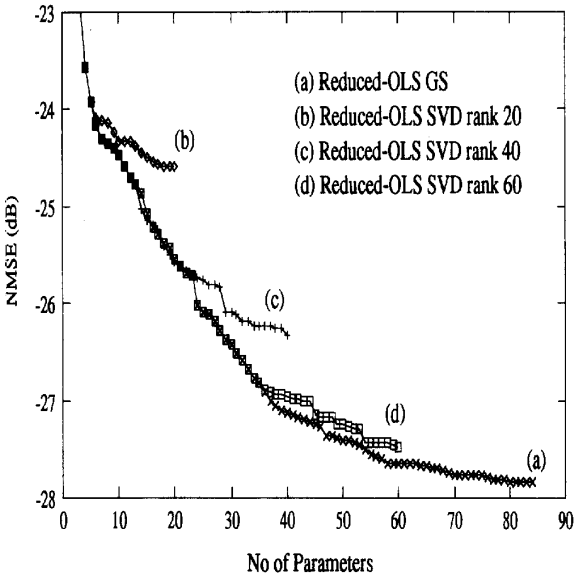

Fig. 3. Performance of subset model found using reduced-OLS on the Volterra predictor.

parameters from an information matrix $\mathbf{X}$ of size $N \times K$:

$$\text{No. multiplications (OLS)} = \sum_{i=1}^{R} (3N(K-i-1)) + \sum_{i=1}^{R-1} (2N(K-i)). \quad (5)$$

The number of multiplications required to perform the preprocessing using the GS decomposition is calculated using

$$\text{No. multiplications (GS decomposition)} = NK^2 + NK. \quad (6)$$

Therefore, the total number of multiplications required to perform a subset selection by the reduced-OLS GS approach is:

$$\text{No. multiplications (reduced-OLS GS)} = NK^2 + NK + \sum_{i=1}^{R} (3K(K-i-1)) + \sum_{i=1}^{R-1} (2K(K-i)). \quad (7)$$

B. Reduced-OLS SVD Approach

To further reduce the computational load of the OLS, we can use an approximated matrix $\hat{\mathbf{X}}$ to represent $\mathbf{X}$. We define $\mathbf{X}$ and $\hat{\mathbf{X}}$ as

$$\mathbf{X} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \quad (8)$$

$$\hat{\mathbf{X}} = \mathbf{U}_\kappa \mathbf{\Lambda}_\kappa \mathbf{V}_\kappa^T \qquad \kappa < K \quad (9)$$

where the columns of $\mathbf{U}$ are the left eigenvectors, $\mathbf{\Lambda}$ is the diagonal matrix containing the singular values and the rows of $\mathbf{V}^T$ are the right eigenvectors formed by using singular value decomposition (SVD) [8] on $\mathbf{X}$. The singular values in $\mathbf{\Lambda}$ are arranged such that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_K$. The $N \times \kappa$ matrix $\mathbf{U}_\kappa$ is formed by using the first κ columns of $\mathbf{U}$, the diagonal $\kappa \times \kappa$ matrix $\mathbf{\Lambda}_\kappa$ is formed by using the first κ rows and columns of $\mathbf{\Lambda}$, and the $\kappa \times K$ matrix $\mathbf{V}_\kappa^T$ is formed by using the first κ rows of $\mathbf{V}^T$. The matrix $\hat{\mathbf{X}}$ is a rank κ approximation of the matrix $\mathbf{X}$ created by the product of $\mathbf{U}_\kappa$, $\mathbf{\Lambda}_\kappa$ and $\mathbf{V}_\kappa^T$.

If $\hat{\mathbf{X}}$ is used to approximate $\mathbf{X}$, (1) can be approximated by

$$\mathbf{y} \approx \hat{\mathbf{X}}\mathbf{h} + \mathbf{e}. \quad (10)$$

Premultiplying the previous equation by $\mathbf{U}_\kappa^T$, we get

$$\mathbf{U}_\kappa^T \mathbf{y} \approx \mathbf{\Lambda}_\kappa \mathbf{V}_\kappa^T \mathbf{h} + \mathbf{U}_\kappa^T \mathbf{e}. \quad (11)$$

If we introduce the $\kappa \times 1$ vectors $\tilde{\mathbf{y}}_\kappa = \mathbf{U}_\kappa^T \mathbf{y}$ and $\tilde{\mathbf{e}}_\kappa = \mathbf{U}_\kappa^T \mathbf{e}$, and the $\kappa \times K$ matrix $\hat{\mathbf{X}}_\kappa = \mathbf{\Lambda}_\kappa \mathbf{V}_\kappa^T$, (11) can be written as

$$\tilde{\mathbf{y}}_\kappa \approx \hat{\mathbf{X}}_\kappa \mathbf{h} + \tilde{\mathbf{e}}_\kappa. \quad (12)$$

Since the dimensions of $\tilde{\mathbf{y}}_\kappa$ and $\hat{\mathbf{X}}_\kappa$ are smaller than those of the vector $\tilde{\mathbf{y}}$ and matrix $\hat{\mathbf{X}}$ in (4), the computation requirement is further reduced when the OLS algorithm is applied. This method is only appropriate when the approximation of $\mathbf{X}$, i.e., $\hat{\mathbf{X}}$, is created by a sufficiently large rank κ, otherwise the subset model found may not be good.

IV. Results of Reduced OLS Methods

Computer simulation was carried out to evaluate the quality of subset models found using the reduced-OLS methods. Subset models were selected from two different 84-tap nonlinear predictors used for predicting a chaotic Mackey-Glass time series. The first predictor was created using a degree 3 and embedding-vector-length 6 Volterra expansion. The second predictor was created by combining a 6-tap linear predictor with a 78-tap RBF predictor.

For the experiment, 500 samples from the Mackey-Glass time series were used to generate the information matrix. The information matrix $\mathbf{X}$ thus had a size of 500 × 84. To measure the modelling quality of the predictor, the normalized mean square error (NMSE) was used:

$$\text{NMSE} = 10 \log_{10} \left(\frac{\sum_{i=1}^{N} e_i^2}{\sum_{i=1}^{N} s_i^2} \right) \quad (13)$$

where s_i is the desired signal value at sample i, and $e_i = s_i - \hat{s}_i$. From (13), we can see that when we have perfect prediction, i.e., $e_i =$

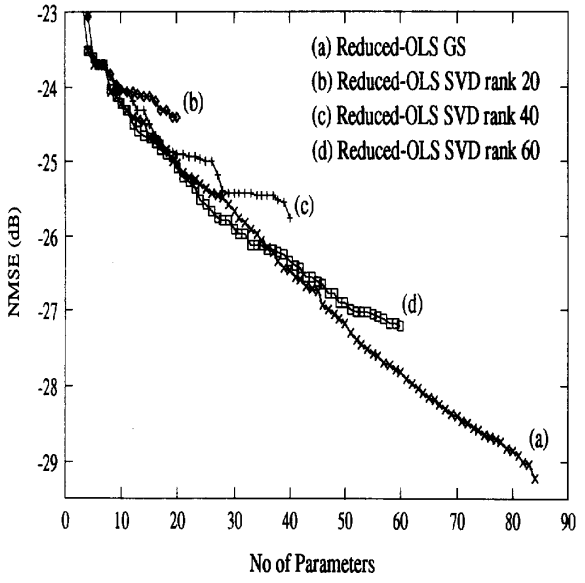

Fig. 4. Performance of subset model found using reduced-OLS on the RBF predictor.

0 for all i, the NMSE will be $-\infty$ dB. When there is no prediction, i.e. $\hat{s}_i = 0$, $e_i = s_i$ for all i, the NMSE will be 0 dB.

The subset models found using the reduced-OLS GS approach were identical to those selected by the OLS using the original data. This, however, is not true for the models found using the reduced-OLS SVD approach. The reason is that the reduced-OLS SVD scheme selected subset models based on $\hat{\mathbf{X}}_\kappa$ and $\hat{\mathbf{y}}_\kappa$, which are approximations of the original data.

Fig. 3 depicts the predictive performance of subset models selected from the Volterra predictor. The results show that when approximation rank $\kappa = 40$ is used, subset models selected with sizes less than 22 have almost equivalent performance to those selected using the full model. This suggests that information regarding the first 22 significant regressors was not lost when we approximated the rank 84 matrix $\mathbf{X}$ by the rank 40 matrix $\hat{\mathbf{X}}_R$. When an approximation rank 60 is used, there is hardly any difference between the subset models selected by the reduced-OLS SVD algorithm and those chosen by the OLS algorithm using the original data.

Similar results were also found for subset models generated from the second nonlinear predictor (Fig. 4). That is, subset models found with large approximation rank κ have very similar predictive performance characteristics to those selected using the original data.

V. Conclusion

A method of reducing computational requirement of the OLS subset model selection algorithm has been presented. This reduction is significant when the number of rows in the information matrix $\mathbf{X}$ is significantly larger than the number of its columns. Two schemes of the reduced-complexity OLS method have been proposed. The first scheme is based on a Gram-Schmidt preprocessing and will provide identical results to those obtained using the original input matrix and the desired output vector. For the second scheme, based on an SVD preprocessing, it has been shown that we can always trade in subset selection performance for computational complexity.

Acknowledgment

The authors thank D. Hughes for providing the Mackey-Glass data used in the simulations.

References

[1] S. Chen, S. A. Billings, and W. Luo, "Orthogonal least squares methods and their application to nonlinear system identification," *Int. J. Contr.*, vol. 50, no. 5, pp. 1873–1896, 1989.

[2] S. Chen, C. F. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Networks*, vol. 2, no. 2, pp. 302–309, 1991.

[3] R. R. Hocking, "The analysis and selection of variables in linear regression," *Biometrics*, vol. 32, pp. 1–49, 1976.

[4] M. J. D. Powell, "Radial basis functions for multivariable interpolation: A review," in *Algorithms for Approximation*, J. C. Mason and M. G. Cox (Eds). Oxford, UK: Oxford University Press, 1987, pp. 143–167.

[5] L. Wang and J. M. Mendel, "Fuzzy basis functions, universal approximation, and orthogonal least-squares learning," *IEEE Trans. Neural Networks*, vol. 3, no. 5, pp. 807–814, 1992.

[6] P. Alper, "A consideration of the discrete Volterra series," *IEEE Trans. Automat. Contr.*, vol. AC-10, pp. 322–327, 1965.

[7] M. J. Korenbergh, "A robust orthogonal algorithm for system identification and time series analysis," *Biolog. Cybern.*, vol. 60, pp. 267–276, 1989.

[8] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD: John Hopkins University Press, 1989, 2nd ed.