AN EFFICIENT ORDER RECURSIVE ALGORITHM FOR VOLTERRA SYSTEM IDENTIFICATION

George-Othon Glentis
TEI of Heraklion, Branch at Chania
Department of Electronics
3, Romanou Str, Halepa, Chania 73133, Greece

Nicholas Kalouptsidis
University of Athens
Department of Informatics
Panepistimiopolis, TYPA Blds, Athens 15784 Greece

ABSTRACT

In this paper nonlinear filtering and identification based on finite support Volterra models is considered. A set of *primary signals*, defined in terms of the input signal, serve for the efficient mapping of the nonlinear process to an equivalent multichannel format. An efficient order recursive method is presented for the determination of the Volterra model structure. The efficiency of the proposed methods is illustrated by simulations.

1 INTRODUCTION

Volterra series modeling is a popular approach to cope with nonlinear time-invariant systems. Volterra models are parametrized with respect to the degree of the nonlinearities and the corresponding memory. Finite support Volterra models are very attractive, since they are generic enough and also reduce to linear regressions. Thus, optimum parameter estimation methods result to the solution of linear systems of equations, both for the Mean Squared Error (MSE) case, as well as for the Least Squares Error (LSE) case. Most methods address this problem by recasting the Volterra filtering and identification model to an equivalent linear multichannel set up. The parameters sought are subsequently obtained by applying linear multichannel parameter estimation algorithms, in batch as well as in adaptive form.

A discrete time finite support regular Volterra model has the form, [1]

$$y(n) = c_0 + \sum_{\ell=1}^{k} \sum_{i_1=0}^{m_\ell-1} \sum_{i_2=0}^{m_\ell-1-i_1} \cdots \sum_{i_\ell=0}^{m_\ell-1-\sum_{j=1}^{\ell-1} i_j} c_\ell(i_1, i_2..i_\ell) \prod_{j=1}^{\ell} x(n - \sum_{q=j}^{\ell} i_q) \tag{1}$$

In the above definition, k is the maximum degree of nonlinearities, while each index m_ℓ, $1 \leq \ell \leq k$, represents the 'memory' of the corresponding nonlinearity. Moreover, $h_\ell(i_1, i_2, \ldots i_\ell)$ is the Volterra kernel of degree ℓ. $x(n)$ is the input and $\eta(n)$ is the unmeasured disturbance.Notice that the triangular and the regular kernels are permutation equivalent, [1, p.246].

The number of coefficients involved into the regular kernel representation is given in terms of the memory associated with each degree of the nonlinearity as $M = 1 + \sum_{\ell=1}^{k} \binom{m_\ell+\ell-1}{\ell}$. For the special case of equal memories for all kernels, we get $M = 1 + \sum_{\ell=1}^{k} \binom{m+\ell-1}{\ell} = \binom{m+k}{k}$.

2 THE MULTICHANNEL EMBEDDING

From the regular formulation of the Volterra model, (1), it follows that the data products involved in the estimation of the filter output possess a shift invariance property. This is also true for the triangular model, subject to a permutation shuffling. Indeed, the data related to the linear part are time shifts of the input signal $x(n)$. The data involved in the second order part are time shifts of the signals $x(n)x(n-i_1)$, $0 \leq i_1 \leq m_2 - 1$, and so on. It is reasonable to define a set of *primary signals* that carry on all the information needed for the estimation of the convolution eq. (1). All other data are produced as time shifts of the primary signals. The primary signals are defined in **Table 1**. The primary signals are embedded into a multichannel signal $\boldsymbol{\phi}_K(n)$

$$\boldsymbol{\phi}_K(n) = [\phi_1(n)\ \phi_2(n)\ \phi_3(n)\ \ \ldots\ \phi_K(n)] =$$
$$[1\ x(n)\ x^2(n)\ x(n)x(n-1) \ldots \prod_{j=1}^{k-1} x(n - \sum_{q=1}^{j} i_q)]^T$$

A multichannel formalism can be introduced for the description of the regular Volterra kernel. The linear term convolves the input signal $x(n)$ with the linear kernel coefficients. The second order term can also be written in a linear regression form, this time however a multichannel formulation is required. It can be written as a linear regression of a multichannel filter with m_2 input signals, namely the primary signals corresponding to the second order term, and filter size m_2 for the first channel up to 1 for the last one. Working in a similar way, the third order term is written as a linear regression of a multichannel filter with $(m_3)(m_3+1)/2$ input signals, and filters size varying from m_3 to 1. In general, the k order kernel of a Volterra filter is written as a multichannel linear regression with $\binom{m_k+k-2}{k-1}$ input signals and filters size varying from m_k to 1.

In this manner, eq. (1) is viewed as a multi-input single-output filter. The number of the primary signals entered the multichannel regression is $K = 2 + \sum_{\ell=2}^{k} \binom{m_\ell+\ell-2}{\ell-1}$. Each filter has size varying form m_1, $m_2 \ldots m_k$ to 1. If $m_i = m_j = m$, then $K = 1 + \binom{m+k-1}{k-1}$.

The linear regression (1) is then described in terms of a multichannel regressor, $y(n) = \mathbf{\Phi}_M^T(n)\boldsymbol{\mathcal{C}}_M + \eta(n)$. $\mathbf{\Phi}_M(n)$ is the multichannel regressor and $\boldsymbol{\mathcal{C}}_M$ is the corresponding filter coefficients vector. Subscript M is utilized to denote that both vectors have dimensions $M \times 1$. $\mathbf{\Phi}_M(n)$ and $\boldsymbol{\mathcal{C}}_M$ are both block vectors that consist of K sub-vectors, of dimensions varying form m_1, $m_2 \ldots m_k$ to 1. The size of each filter can be retrieved from a multi index that carries all the filters' size for each primary signal associated to the Volterra kernels. To this end, let us define the multi index

$$P = [\ \underbrace{1}_{0th-kernel}\ \ \underbrace{p^1}_{1th-kernel}\ \ \underbrace{p_0^2 \cdots p_{m_2-1}^2}_{2nd\ kernel} \cdots \underbrace{p_{0\ldots 0}^k \cdots p_{m_k-1,m_2-1,\ldots m_k-1}^\ell}_{kth\ kernel}] \tag{2}$$

Clearly, P has K elements, equal to the number of the primary signals, and each element denotes the filter size corresponding to a primary signal. Moreover, $1 \leq p_{i_1,i_2\ldots i_\ell}^\ell \leq m_\ell$, and M, the number of the filter coefficients given is now alternatively estimated as $M = \sum_{i=1}^K P(i)$. Thus, the regressor vector is written in a block vector form, as

$$\mathbf{\Phi}_M(n) = [\,1\ \mathbf{\Phi}_{P(2)}^T(n)\ \mathbf{\Phi}_{P(3)}^T(n)\ \cdots\ \mathbf{\Phi}_{P(K)}^T(n)\ ]^T \tag{3}$$

where each ubvector $\mathbf{\Phi}_{P(i)}(n)$ carries the data associated with a primary signal, i.e.,

$$\mathbf{\Phi}_{P(i)}(n) = [\phi_i(n)\ \phi_i(n-1)\ \ldots \phi_i(n-P(i)+1)\ ]^T$$

and $P(i)$ is the corresponding filter size as is assigned in the multi-index P.

The Volterra regression that minimizes the total squared error $E_M(N) = \sum_{n=0}^N (y(n) - \mathbf{\Phi}_M^T(n)\boldsymbol{\mathcal{C}}_M(n))^2$, is seeked. The resulting normal equations have the form

$$\boldsymbol{\mathcal{R}}_M(N)\boldsymbol{\mathcal{C}}_M(N) = \boldsymbol{\mathcal{D}}_M(N) \tag{4}$$

where

$$\boldsymbol{\mathcal{R}}_M(N) = \sum_{n=0}^N \mathbf{\Phi}_M(n)\mathbf{\Phi}_M^T(n), \quad \boldsymbol{\mathcal{D}}_M(n) = \sum_{n=0}^N \mathbf{\Phi}_M(n)y(n) \tag{5}$$

Eq. (4) corresponds to a block near to Toeplitz linear system,[5]. Following [5], efficient recursions for estimating the optimum filter $\boldsymbol{\mathcal{C}}_{M+1}(N)$ from the lower order counterpart $\boldsymbol{\mathcal{C}}_M(N)$ have been developed in [9]. The resulting algorithm is tabulated on **Table 2**. The computational cost for the order updating procedure is $O(KM)$. In a similar way, an order downdating can be derived, [9], see **Table 3**. The order updating and the order downdating algorithms of Tables 2 and 3, are utilzed in the sequel to form an efficient order searching scheme.

3 VERSATILE ORDER RECURSIVE ALGORITHMS

The problem of identifing the structure of a linear or a nonlinear system, parametrized by linear or nonlinear polynomial models, has long been studied in the context of the identification for prediction and control, [6]-[8]. In our case, this is casted as follows: given the maximum degree of the nonlinearities k, as well as the maximum memory size associated to each kernel, m_ℓ, $0 \leq \ell \leq k$, find the subset of coefficients that specifies the structure of the system, in some optimum way. Consider the regular model of eq. (1). The maximum number of parameters addressed by this model is M. The optimum filter is then estimated by the LS solution, eq. (4), and the minimum LSE error attained is given by

$$E_M(N) = \sum_{n=0}^N y^2(n) - \boldsymbol{\mathcal{D}}_M^T(N)\boldsymbol{\mathcal{C}}_M(N) \tag{6}$$

In order to find the best Volterra structure, all models corresponding to all possible combinations of the index space should be tested. Exhaustive search requires some 2^{M-1} trials. The cost of such task is heavy and grows up exponentially even for the linear part of the model. Suboptimal algorithms that pick up a single coefficient at a time have been proposed.

To overcome the engagement with a huge amount of candidate models, the following assumption is imposed: The model set consists of hierarchical models, with respect to the evolution of the memory associated with each channel. We are going to describe a rather simple approach, yet effective for a certain class of problems where the index space is restricted to evolve (grow up) in a smooth way, i.e., index gaps are not allowed within each kernel. Then, the Volterra structure is described by the multi-index P defined by eq. (2).

Suppose that the optimum filter $\boldsymbol{\mathcal{C}}_M(N)$ has been estimated. It corresponds to the multi-index P and has dimensions $M \times 1$. The proposed order updating procedure results to a new filter with $M+1$ coefficients and may come in one of the following three forms: **a)** increase the memory corresponding to a primary signal of a certain nonlinearity, **b)** introduce a new primary signal associated with a nonlinear kernel, and **c)** introduce a new nonlinearity.

Similarly, we may apply an order downdating scheme to delete of a filter coefficient, i.e., decrease the number of the filter taps from M to $M-1$. This can be done by one of the following three ways: **a)** decrease the memory corresponding to a primary signal of a certain nonlinearity, **b)** remove a single coefficient primary signal associated with a nonlinear kernel, and **c)** remove a single coefficient nonlinearity.

With the above order updating schemes at our disposal, a (suboptimal) order searching algorithm can be applied as follows **1)** Given $\boldsymbol{\mathcal{C}}_M(N)$ of size $M \times 1$, estimate $\boldsymbol{\mathcal{C}}_{M+1}(N)$ of size $(M+1) \times 1$, by increasing the number of coefficients by one, following each one of the three ways discussed. Select the one that leads to the minimum LSE error. The search is conducted over K candidate filters. **2)** Among all filters of size $(M+1)\times 1$ that can be produced from $\boldsymbol{\mathcal{C}}_{M+1}(N)$ by a simultaneous increase of one channel size by a adding a single coefficient, while reducing another channel size by deleting a single coefficient (thus the size of the filter remains the same), select the one that leads to the minimum LSE error. The search is conducted over $K(K-1)$ candidate filters. **3)** Iterate on step $2I$ times, I being a user specified integer. **4)** Iterate on step 1 until a predefined minimum LSE error is attained, or the filter reaches at a maximum size.

Order estimation indices can be utilized for the location of the best model, that combines a minimum fitting error, with the lowest number of parameters. The AIC_w and the BIC are two popular criteria that can directly be applied based on the minimum squared error $E_M(N)$ and the number of processed data N, [6]-[8]. They have the form

$$AIC_w(M) = N\ln(E_M(N)/(N-M)) + wM, \quad w>0$$
$$BIC(M) = N\ln(E_M(N)/(N-M)) + M\ln(N)$$

4 SIMULATIONS

The performance of the search engine described in Section 7, using the versatile order recursive algorithm, was tested for a second order Volterra filter, with $k=2$, and memory sizes $m_1 = 5$ and $m_2 = 5$. Thus, the number of channels is $K = 7$, and the number of coefficients is $M = 21$. The structure of this Volterra system, is described by the multi-index

$$P = [\ \underbrace{1}_{0th-kernel}\quad \underbrace{p^1}_{1st-kernel}\quad \underbrace{p_0^2\ p_2^2\ p_3^2\ p_4^2\ p_4^2}_{2nd-kernel}]$$
$$[\ \underbrace{1}_{0th-kernel}\quad \underbrace{5}_{1st-kernel}\quad \underbrace{5\ 4\ 3\ 2\ 1}_{2nd-kernel}]$$

The algorithm was allowed to search for the best filter, of maximum size $M = 30$, with $K = 10$ allowable channels, one for the constant, one for the linear part, and 8 for the second order part. The corresponding primary signals were ϕ_1, $\phi_2(n) = x(n)$, and $\phi_{2+i+1}(n) = x(n)x(n-i)$, $i = 0, 1 \ldots 7$. A white noise input signal of size $N = 512$ was tested. The SNR was set to 20 db. The minimum squared error attained, as well as the AIC_2 and BIC curves are shown in **Figure 1** .The correct structure of the system was predicted.

5 CONCLUSIONS

Nonlinear filtering and identification based on finite support Volterra models has been considered in this paper. A unified framework for the multichannel embedding of the Volterra system has been proposed. The regular Volterra model has been utilized to facilitate the passage from a single input single output finite support Volterra model to a multi input single output linear regression. A set of primary signals that carries products of input data has been introduced. These auxiliary signals serve as inputs to the multichannel linear regression and possess shift invariance properties. An order recursive scheme capable of searching for the best model structure has been derived, for the case when the Volterra model structure is not known in advance. Efficient order increasing or decreasing algorithms have been derived to faciliate as fast computational engines during the search for the best model fit. The efficiency of the proposed methods has been illustrated by simulations.

References

[1] W.J. Rugh, Nonlinear System Theory, John Hopkins Univ. Press, 1981

[2] N. Kalouptsidis, 'Signal Processing Systems: Theory and Design', John Wiley 1997

[3] V. J. Mathews, 'Adaptive polynomial filters, 'IEEE Signal Proc. Mag., pp. 10-26, July 1991.

[4] N. Kalouptsidis, G. Glentis and P. Koukoulas, Efficient algorithms for parameter Volterra system identification, EUSIPCO-94, Edinburgh, pp. 212-215.

[5] G. Glentis and N. Kalouptsidis, Efficient multichannel FIR filtering using a step versatile order recursive algorithm, Signal Processing, vol 37, No 3, pp. 437-462, June 1994.

[6] M. Haest, G. Bastin, M. Gevers, and V. Wertz, ' EPSION: an expert system for system identification,' Automatica, vol. 26, no. 1, pp. 85-96, 1990.

[7] R. Haber, and H. Unbehauen, Structure identification of nonlinear dynamic systems- A survey of input/output approaches,' Automatica, vo. 26, no. 4, pp. 651-677, 1990.

[8] M. Pottmann, H. Unbehauen, and D. Seborg, 'Application of a general multi-model approach for system identification of highly nonlinear processes- a case study,' Int. J. Control, vol. 57, no. 1, pp. 97-120, 1993.

[9] G. Glentis, P. Koukoulas and N. Kalouptsidis, 'Efficient algorithms for Volterra system identification, submitted for publication in the IEEE Trans. on Signal Processing, Dec. 1997.

$Primary Signal$	$Index Range$
1	
$x(n)$	
$x(n)x(n-i_1)$	$0 \le i_1 \le m_2 - 1$
$x(n)x(n-i_2)x(n-i_1-i_2)$	$0 \le i_1 \le m_3 - 1,$ $0 \le i_2 \dots m_3 - 1 - i_1$
$\dots \quad \dots$	.
$\prod_{j=1}^{k-1} x(n - \sum_{q=j}^{\ell} i_{i_q})$	$0 \le i_1 \le m_k - 1,$ $0 \le i_2 \le m_k - 1 - i_1,$ $\dots$ $0 \le i_{k-1} \le m_k - 1 - \sum_{q=1}^{j} i_q$

Table 1. The primary signals

$$\beta^{c}_{M}(i)(N) = d_i(N) + \mathcal{R}^{b(i)T}_{M}(N)\boldsymbol{\mathcal{C}}_M(N)$$
$$\alpha^{b(i)}_{M}(N) = r^{bo}_{i}(N) + \mathcal{R}^{b(i)T}_{M}(N)\boldsymbol{\mathcal{B}}^i_M(N)$$
$$k^{c(i)}_{M}(N) = -\beta^{c(i)}_{M}(N)/\alpha^{b(i)}_{M}(N)$$
$$S^i\ \boldsymbol{\mathcal{C}}_{M+1}(N) = \begin{bmatrix} \boldsymbol{\mathcal{C}}_M(N) \\ 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{\mathcal{B}}^i_M(N) \\ 1 \end{bmatrix} k^{c(i)}_M(N)$$
$$\epsilon^{b(i)}_M(N) = \phi_i(N - P(i)) + \boldsymbol{\Phi}^T_M(N)\boldsymbol{\mathcal{B}}^i_M(N)$$
$$\boldsymbol{\mathcal{B}}^i_M(N-1) = \boldsymbol{\mathcal{B}}^i_M(N) - \boldsymbol{\mathcal{W}}_M(N)\epsilon^{b(i)}_M(N)$$
$$e^{b(i)}_M(N) = \epsilon^{b(i)}_M(N)\alpha_M(N)$$
$$\alpha^{b(i)}_M(N-1) = \alpha^{b(i)}_M(N) - \epsilon^{b(i)}_M(N)e^{b(i)}_M(N)$$
$$k^{w}_{M}(N) = -e^{b(i)}_M(N)/\alpha^{b(i)}_M(N-1)$$
$$S^i\ \boldsymbol{\mathcal{W}}_{M+1}(N) = \begin{bmatrix} \boldsymbol{\mathcal{W}}_M(N) \\ 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{\mathcal{B}}^i_M(N-1) \\ 1 \end{bmatrix} k^{w}_M(N)$$
$$\alpha_{M+1}(N) = \alpha_M(N) + e^{b(i)}_M(N)k^w_M(N)$$
$$\textbf{FOR} \quad j = 1, 2 \dots K \quad \textbf{AND} \quad j \ne i \quad \textbf{DO}$$
$$\beta^{b(j)}_M(N) = r^{b(j)}_i(N) + \boldsymbol{\mathcal{B}}^{jT}_M(N)\mathcal{R}^{b(i)}_M(N)$$
$$k^{b(j)}_M(N) = -\beta^{b(j)}_M(N)/\alpha^{b(i)}_M(N)$$
$$S^i\ \boldsymbol{\mathcal{B}}^j_{M+1}(N) = \begin{bmatrix} \boldsymbol{\mathcal{B}}^j_M(N) \\ 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{\mathcal{B}}^i_M(N) \\ 1 \end{bmatrix} k^{b(j)}_M(N)$$
$$\textbf{END} \quad j$$
$$\textbf{IF} \quad j = i \quad \textbf{THEN} \quad \textbf{DO}$$
$$\boldsymbol{\beta}^{b(i)}_{M+k-1}(N) = \bar{\mathrm{r}}^{b(i)}_{k-1}(N) + \boldsymbol{\mathcal{A}}^T_M(N)\bar{\mathcal{R}}^{bi}_M(N-1)$$
$$\bar{\mathrm{k}}^{b(i)}_{M+K-1}(N) = -\alpha^{-f}_M(N)\boldsymbol{\beta}^{b(i)}_{M+K-1}(N)$$
$$\mathbf{T}\boldsymbol{\mathcal{B}}^i_{M+K-1}(N) = \begin{bmatrix} \mathbf{0}_{K\times 1} \\ \boldsymbol{\mathcal{B}}^i_M(N-1) \end{bmatrix} + \begin{bmatrix} \mathbf{I}_{K\times K} \\ \boldsymbol{\mathcal{A}}_M(N) \end{bmatrix} \bar{\mathrm{k}}^{b(i)}_{M+K-1}(N)$$
$$\mathbf{S}_i\boldsymbol{\mathcal{B}}^i_{M+K-1}(N) = \begin{bmatrix} \mathcal{X}_{M+1} \\ \tilde{\mathrm{k}}^{b(i)}_{M+K-2}(N) \end{bmatrix}$$
$$\tilde{\boldsymbol{\mathcal{B}}}_{M+1}(N) = [\boldsymbol{\mathcal{B}}^1_{M+1}(N)$$
$$\dots\ \boldsymbol{\mathcal{B}}^{(i-1)}_{M+1}(N)\ \boldsymbol{\mathcal{B}}^{(i+1)}_{M+1}(N)\ \dots\ \boldsymbol{\mathcal{B}}^K_{M+1}(N)\ ]$$
$$\boldsymbol{\mathcal{B}}^i_{M+1}(N) = \mathcal{X}_{(M+1)} - \tilde{\boldsymbol{\mathcal{B}}}_{M+1}(N)\tilde{\mathrm{k}}^{b(i)}_{M+K-2}(N)$$
$$\textbf{ENDIF}$$
$$k^{f(i)}_M(N) = -\beta^{b(i)T}_M(N)/\alpha^{bi}_M(N-1)$$
$$S_i\ \boldsymbol{\mathcal{A}}_{M+1}(N) = \begin{bmatrix} \boldsymbol{\mathcal{A}}_M(N) \\ 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{\mathcal{B}}^i_M(N-1) \\ 1 \end{bmatrix} k^{f(i)}_M(N)$$
$$\alpha^{-f}_M(N) = \alpha^{-f}_M(N) - \frac{\alpha^{-f}_M(N)\beta^{bT}_M(N)(\alpha^{-f}_M(N)k^b_M(N))^T}{1+\beta^{bt}_M(N)\alpha^{-f}_M(N)k^b_M(N)}$$

Table 2. The versatile order recursive algorithm. Forward recursions

$$\beta^{b(i)}_{K-1}(N) = \tilde{r}^{b(i)}_{K-2}(N) + \tilde{\mathcal{R}}^{bT}_{M+1}(N)\boldsymbol{\mathcal{B}}^i_{M+1}(N)$$
$$\alpha^{b(i)}_{K-1}(N) = \mathrm{r}^{bo}_{K-2}(N) + \tilde{\mathcal{R}}^{bT}_{M+1}(N)\tilde{\boldsymbol{\mathcal{B}}}_{M+1}(N)$$
$$\tilde{k}^{b(i)}_{M+1}(N) = -\alpha^{-b(i)}_{K-1}(N)\beta^{b(i)}_{K-1}(N)$$
$$\mathbf{S}\boldsymbol{\mathcal{B}}^i_{M+K-1}(N) =$$
$$\begin{bmatrix} \boldsymbol{\mathcal{B}}^i_{M+1}(N) \\ \mathbf{0}_{(K-2)\times 1} \end{bmatrix} + \begin{bmatrix} \tilde{\boldsymbol{\mathcal{B}}}_{M+1}(N) \\ \mathbf{I}_{(K-2)\times(K-2)} \end{bmatrix} \tilde{k}^{b(i)}_{M+1}(N)$$
$$\mathbf{T}\boldsymbol{\mathcal{B}}^i_{M+K-1}(N) = \begin{bmatrix} \bar{k}^{b(i)}_{M+1}(N) \\ \bar{\boldsymbol{\mathcal{B}}}_{M+1}(N) \end{bmatrix}$$
$$S_i\ \boldsymbol{\mathcal{A}}_{M+1}(N) = \begin{bmatrix} \bar{\boldsymbol{\mathcal{A}}}_{M+1}(N) \\ \bar{k}^f_{M+1}(N) \end{bmatrix}$$
$$\boldsymbol{\mathcal{A}}_M(N) = (\bar{\boldsymbol{\mathcal{A}}}_{M+1}(N) - \bar{\boldsymbol{\mathcal{B}}}_{M+1}(N).$$
$$.\bar{k}^f_{M+1}(N))(\mathbf{I}_{K-1} - \bar{k}^f_{M+1}(N)\bar{k}^{b(i)}_{M+1}(N))^{-1}$$
$$\boldsymbol{\mathcal{B}}_M(N-1) = (\bar{\boldsymbol{\mathcal{B}}}_{M+1}(N) - \bar{\boldsymbol{\mathcal{A}}}_{M+1}(N).$$
$$.\bar{k}^{b(i)}_{M+1}(N))/(1 - \bar{k}^{b(i)}_{M+1}(N)\bar{k}^f_{M+1}(N))$$
$$S^i\ \boldsymbol{\mathcal{W}}_{M+1}(N) = \begin{bmatrix} \bar{\boldsymbol{\mathcal{W}}}_{M+1}(N) \\ k^w_M(N) \end{bmatrix}$$
$$\boldsymbol{\mathcal{W}}_M(N) = \bar{\boldsymbol{\mathcal{W}}}_{M+1}(N) - \boldsymbol{\mathcal{B}}_M(N-1)k^w_M(N)$$
$$\alpha_M(N) = \alpha_{M+1}(N) - e^{b(i)}_M(N)k^w_M(N)$$
$$e^{b(i)}_M(N) = \phi_i(N - P(i)) + \boldsymbol{\Phi}^T_M(N)\boldsymbol{\mathcal{B}}^i_M(N-1)$$
$$\epsilon^{b(i)}_M(N) = e^{b(i)}_M(N)/\alpha_M(N)$$
$$\boldsymbol{\mathcal{B}}^i_M(N) = \boldsymbol{\mathcal{B}}^i_M(N-1) + \boldsymbol{\mathcal{W}}_M(N)\epsilon^{b(i)}_M(N)$$
$$S_i\boldsymbol{\mathcal{C}}_{M+1} = \begin{bmatrix} \bar{\boldsymbol{\mathcal{C}}}_{M+1}(N) \\ k^{c(i)}_M(N) \end{bmatrix}$$
$$\boldsymbol{\mathcal{C}}_M(N) = \bar{\boldsymbol{\mathcal{C}}}_{M+1}(N) - \boldsymbol{\mathcal{B}}^i_M(N)k^{c(i)}_M(N)$$
$$\textbf{FOR} \quad j = 1, 2 \dots K \quad \textbf{AND} \quad j \ne i \quad \textbf{DO}$$
$$S_i\boldsymbol{\mathcal{B}}^j_{M+1} = \begin{bmatrix} \bar{\boldsymbol{\mathcal{B}}}^j_{M+1}(N) \\ k^{b(j)}_M(N) \end{bmatrix}$$
$$\boldsymbol{\mathcal{B}}^j_M(N) = \bar{\boldsymbol{\mathcal{B}}}^j_{M+1}(N) - \boldsymbol{\mathcal{B}}^i_M(N)k^{b(j)}_M(N)$$
$$\textbf{END} \quad j$$

Table 3. Efficient decreasing recursions for the versatile order recursive algorithm.

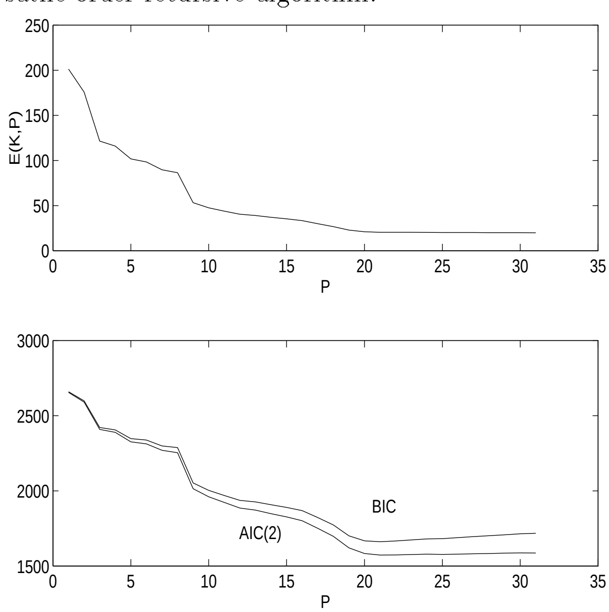

Figure 1: The order search