

# FAST ORDER-RECURSIVE ALGORITHMS FOR OPTIMAL STACK FILTER DESIGN

Ioan Tăbuș<sup>1</sup> and Moncef Gabbouj

Signal Processing Laboratory  
Tampere University of Technology  
P.O.Box 553, FIN-33101 Tampere, Finland  
tabus@cs.tut.fi, moncef@cs.tut.fi

## ABSTRACT

For the nonlinear classes of filters based on the median archetype, e.g. stack, Weighted Order Statistics (WOS), morphological filters the techniques exploiting recursiveness of embedded structures were not used yet.

We investigate in this paper the possibilities of improving the speed of the optimal design techniques restating the optimal design problem as a sequence of optimal problems for embedded structures.

The speed up effect of recursive-in-order design techniques is very significant but it is not the only benefit of this principle: it also allows the evaluation of the *optimal structure* of the filter, comparing the performances of the optimal filters having various structural parameters and observing where the performance index curve starts to flatten with the increase in the structure "size".

## 1. INTRODUCTION

The linear filtering literature is very rich in techniques used for exploiting the recursiveness of the model parameter estimates for embedded structures.

In the nonlinear filtering based on polynomial Volterra approximations, the recursive in order structure plays a significant role in improving the efficiency of the estimation algorithms which would otherwise require a high computational effort.

For the stack type nonlinear filters the "order" of the filter can be defined as the parameter(s) which characterize the shape of the processing window around the current sample while the set of parameters, which can be modified when the shape of the window is fixed, will be called model parameters. These algorithms fall

inside two main categories: optimal design in the binary domain (reducing the integer signal case to the binary one through threshold decomposition) and direct design in the integer domain using various search strategies (usually based on gradient techniques).

We investigate here the algorithmic aspects of setting in an embedded structure the problem of optimal nonlinear Boolean and stack filter design in the threshold domain. We also give examples showing the solution of *optimal structure selection* problem which can be seen as a side benefit, virtually with no extra-cost, of the current optimal design recursive in order procedures.

## 2. OPTIMAL BOOLEAN FILTER DESIGN

We elaborate here on the procedure presented in [5],[6] for computing the cost coefficients associated with the MAE of the signals modelled by a Markov chain.

The number of operations required for computing the cost coefficients  $\underline{c}_{N,N_c}$  with the fixed order algorithm is  $\mathcal{O}(2^N N n^2)$ , while using the order recursive algorithm we will obtain all the cost coefficients

$$\underline{c}_{N_c+1,N_c}, \dots, \underline{c}_{N,N_c}$$

with only  $\mathcal{O}(2^{N+1} n^2)$  operations (the integer  $N_c$  defines the position of the current sample inside the  $N$ -length processing window).

The Markov chain (MC) used to model the signals has a finite state space  $Q = \{q_1, q_2, \dots, q_n\}$  and a transition probability matrix  $P$ , with elements  $P(i, j) = \text{Prob}(q(t+1) = q_j | q(t) = q_i)$ . The desired (original) signal  $S(t)$  and the perturbed signal  $X(t)$  are deterministic functions of state:

$$f^S : Q \rightarrow \{0, 1\} \quad ; \quad S(t) = f^S(q(t + N_c))$$

<sup>1</sup>On leave from Polytechnic University of Bucharest, Department of Control and Computers

$$f^X : Q \rightarrow \{0, 1\} ; X(t) = f^X(q(t + N_c)) \quad (1)$$

The following vectors will be used in the sequel (note that all vectors will be underlined):

$$\begin{aligned} \underline{f}^S &= [f^S(q_1), \dots, f^S(q_n)]^T \\ \underline{f}^X &= [f^X(q_1), \dots, f^X(q_n)]^T \end{aligned} \quad (2)$$

The limiting probabilities of the Markov chain,  $\underline{\pi} = [\pi(q_1), \pi(q_2), \dots, \pi(q_n)]^T$ , result as the solution of the system

$$\underline{\pi} = P^T \underline{\pi}, \quad \sum_{i=1}^n \pi(q_i) = 1, \quad (3)$$

and correspond to the stationary probability distribution of the states of the MC model.

The structure of the filter is determined by the processing window of the filter, which will be denoted by

$$\begin{aligned} \underline{X}(t)_{1:m} &= [X(t+1-N_c), \dots, X(t+m-N_c)] \\ &= [f^X(q(t+1)), \dots, f^X(q(t+m))] \end{aligned} \quad (4)$$

(the subscript  $1:m$  will be used to specify the length,  $m$ , of the vector, while the subscript  $k$  will specify the  $k$ -th entry of the vector).

**Problem** *Optimality problem for embedded Boolean and stack filter structures*

Given

- the Markov chain  $(P, \underline{f}^X, \underline{f}^S)$ ;
- the structural parameter,  $m$ , of the embedded filtering windows,  $[\underline{X}]_{1:m}$ ;
- the fixed integer  $N_c$  specifying the position of the current desired signal, relative to the filtering window;

find the optimal Boolean or stack filter, determined by the Boolean function  $g_m^*$ , for each window structure, minimizing the criterion

$$J_m(g_m) = E|S(t) - g_m(\underline{X}(t)_{1:m})| \quad (5)$$

□

Denote by  $\underline{k}_{1:m}$  the  $m$ -dimensional vector having as entries the bits of the binary representation of the integer  $k$  when using  $m$  bits ( $\underline{k}_1$  is the most significant bit and  $\underline{k}_m$  is the least significant bit). The criterion (5) can be rewritten in the form (see [1],[6])

$$J_m(g_m) = C_{0_m} + \sum_{i=0}^{2^m-1} c_{m,N_c}(\underline{i}_{1:m}) g_m(\underline{i}_{1:m}) \quad (6)$$

where the cost coefficients are given by

$$\begin{aligned} c_{m,N_c}(\underline{i}_{1:m}) &= \\ &= \text{Prob}(\underline{X}(t)_{1:m} = \underline{i}_{1:m}, S(t) = 0) \\ &- \text{Prob}(\underline{X}(t)_{1:m} = \underline{i}_{1:m}, S(t) = 1) \end{aligned} \quad (7)$$

for all integers  $i = 0, \dots, 2^m - 1$ . The solution of the embedded problem will be shown to be more effective than solving individually each optimal problem, due to the order recursive procedure for the cost coefficients, which is the main computational task in deriving the optimal filters.

In order to derive recursive formulas, some auxiliary variables will be introduced (which are similar to standard variables used in Hidden Markov Model estimation algorithms [3]):

$$\begin{aligned} \underline{\alpha}_m(\underline{i}_{1:m}) &= \\ &= \begin{bmatrix} \text{Prob}(\underline{X}(t)_{1:m} = \underline{i}_{1:m}, q(t+m) = q_1) \\ \vdots \\ \text{Prob}(\underline{X}(t)_{1:m} = \underline{i}_{1:m}, q(t+m) = q_n) \end{bmatrix} \end{aligned} \quad (8)$$

When the probabilities in (8) are computed for the same joint events, but including also  $S(t) = f^S(q(t + N_c)) = 0$  (or  $S(t) = f^S(q(t + N_c)) = 1$ ), corresponding to the joint limiting probabilities, the auxiliary variables will be denoted by  $\underline{\alpha}_{m,N_c}^0$  (or  $\underline{\alpha}_{m,N_c}^1$ ). The last auxiliary variable is

$$\underline{\alpha}_{m,N_c}^\Delta(\underline{i}_{1:m}) = \underline{\alpha}_{m,N_c}^0(\underline{i}_{1:m}) - \underline{\alpha}_{m,N_c}^1(\underline{i}_{1:m}) \quad (9)$$

which can be used to compute the cost coefficients as follows

$$c_{m,N_c}(\underline{i}_{1:m}) = \underline{1}_{1:m}^T \underline{\alpha}_{m,N_c}^\Delta(\underline{i}_{1:m}) \quad (10)$$

which is a straightforward restating of (7) ( $\underline{1}$  denotes the vector having all the entries equal to 1). The following notation will be useful for stating the recursions for the auxiliary variables:

$$\underline{v} \circ b = b\underline{v} + \bar{b}\bar{\underline{v}}, \quad (11)$$

(logic coincidence function between the vector  $\underline{v}$  and the binary scalar  $b \in \{0, 1\}$ ), and we will denote

$$P^{[b]} = \text{diag}[\underline{f}^X \circ b] P^T \quad (12)$$

*Lemma*

1. The following recursions hold true

$$\begin{aligned} \underline{\alpha}_{m+1}([\underline{x}_{1:m} \ b]) &= P^{[b]} \underline{\alpha}_m(\underline{x}_{1:m}) \\ \underline{\alpha}_{m+1,N_c}^\Delta([\underline{x}_{1:m} \ b]) &= P^{[b]} \underline{\alpha}_{m,N_c}^\Delta(\underline{x}_{1:m}) \end{aligned} \quad (13)$$

for all binary vectors  $\underline{x}_{1:m}$  and binary scalars  $b$ . The first recursion holds for all  $m$ , but the second only for  $m > N_c$ .

2. For  $m = N_c$ , the connection between the auxiliary variables is

$$\underline{\alpha}_{N_c, N_c}^{\Delta}(\underline{i}_{1:N_c}) = (\text{diag}[\underline{f}^S \circ 0] - \text{diag}[\underline{f}^S \circ 1])\underline{\alpha}_{N_c} \quad (14)$$

3. The initialization of the recursions is accomplished taking  $\underline{\alpha}_1(1) = \text{diag}[\underline{f}^X \circ 1]\underline{\pi}$  and  $\underline{\alpha}_1(0) = \text{diag}[\underline{f}^X \circ 0]\underline{\pi}$ .

□ The proof is omitted due to space limitation.

Now let us group in matrices the auxiliary vectors for all integers  $i = 0, \dots, 2^m - 1$

$$\alpha_m = [\underline{\alpha}_m(\underline{0}_{1:m}) \dots \underline{\alpha}_m(\underline{(2^m - 1)}_{1:m})] \quad (15)$$

and similarly define  $\alpha_{m, N_c}^{\Delta}$ .

Then straightforward computations lead to the formulas presented algorithmically in Table 1. This algorithm exploits the "right concatenation" recursive structure of the processing window  $\underline{x}_{1:m+1} = [\underline{x}_{1:m} b]$  used in (13). Since a recursion similar to (13) can be obtained for "left concatenation" structures, various algorithms can be obtained, using combinations of the *right* and *left* enlarging of the processing window (which will correspond to different descending paths in the tree in Table 2).

### 3. OPTIMAL STACK FILTER DESIGN

The cost coefficients found using the algorithm in Table 1 can now be used to design the optimal stack filter using the stacking matrix concept presented in [4]. The modular and recursive in order structure of the stacking matrix can be efficiently combined with the stages Table 1 to obtain a procedure which is recursive-in-order for stack filter design.

### 4. STRUCTURE SELECTION USING THE RECURSIVE IN ORDER PROCEDURES

There is an intensive effort in solving theoretically the problem of best structure selection for embedded structure in the signal processing area.

The basic rationale under most of the methods proposed is to combine the performance criterion which must be minimized at each order stage of the optimal design (criterion which is obviously decreasing with the increase of the "order") with a penalty term which is increasing with the order.

This is also equivalent to finding a way to evaluate when the decrease in the performance criterion is too

small to justify the increasing of the cost due to the use of a larger filter structure.

We illustrate by means of an example the ease of structure selection when using the fast order-recursive procedures.

In Table 2 we show the performance criterion for embedded structures starting from a window of size 3 samples around the central sample and ending with structures having 7 samples. It is obvious that the best structure for the optimal filter is (5,3) (5 samples, the current sample in the middle), because enlarging it, no increase in the performance cost can be gained, in the case of stack filters.

The saturation effect in increasing the size of the window is obvious. Also it is interesting to note the effect on the criterion value caused by the position of the current pixel inside the processing window.

### 5. REFERENCES

- [1] E. J. Coyle and J.-H. Lin. Stack filters and the mean absolute error criterion. *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-36:1244-1254, Aug. 1988.
- [2] M. Gabbouj and E.J. Coyle. Minimax stack filtering in a parameterized environment. In *Proceedings IEEE International Symposium on Circuits and Systems*, pages 97-100, Sheraton Harbour Island, San Diego, USA, May 10-13 1992.
- [3] R. L. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257-286, February 1989.
- [4] I. Tăbuş, and M. Gabbouj. Stacking matrix based fast procedure for optimal stack filter design. In *Proc. SPIE Image Algebra and Morphological Image Processing*, San Diego, July 1994.
- [5] I. Tăbuş, D. Petrescu, and M. Gabbouj. A training framework for Boolean and stack filtering: Optimal design and robustness case studies. Technical Report, 71 pages ISBN 951-722-036-7, Signal Processing Laboratory, Tampere University of Technology, Tampere, Finland, Aug 1993.
- [6] I. Tăbuş, D. Petrescu, and M. Gabbouj. A training framework for Boolean and stack filtering, Part II: Robustness case studies. *IEEE Transactions on Image Processing Special Issue on Nonlinear Image Processing*, submitted.

0.	$\alpha_1 = \begin{bmatrix} \text{diag}[\underline{f}^X \circ 0] \underline{x} & \text{diag}[\underline{f}^X \circ 1] \underline{x} \end{bmatrix}$
1. for $m = 2$ to $N$	
1.1 if $(m \leq N_c)$ then	$\alpha_m = [P^{[0]} \alpha_{m-1} \quad P^{[1]} \alpha_{m-1}]$
1.2 if $(m = N_c)$ then	$\alpha_{m,N_c}^{[\Delta]} = (I - \text{diag}(\underline{f}^S)) \alpha_{m,N_c}$
1.3 if $(m > N_c)$ then	
1.3.1	$\alpha_{m,N_c}^{[\Delta]} = [P^{[0]} \alpha_{m-1,N_c}^{[\Delta]} \quad P^{[1]} \alpha_{m-1,N_c}^{[\Delta]}]$
1.3.2	$\underline{c}_{m,N_c} = \underline{1}^T \alpha_{m,N_c}^{[\Delta]}$
1.3.3 Optimal Boolean filter of "order" $m$ (in Matlab <sup>®</sup> notation)	
	$\underline{g}_{m,N_c} = (\underline{c}_{m,N_c} < 0)$

Table 1: Fast Order Recursive Cost Coefficient Computation and Optimal Boolean Filter Design

				(3,2) 0.0932 0.0932				
			(4,2) 0.0932 0.0932		(4,3) 0.0932 0.0932			
		(5,2) 0.0802 0.0979		(5,3) 0.0815 0.0904		(5,4) 0.0802 0.0979		
	(6,2) 0.0743 0.0998		(6,3) 0.0730 0.0904		(6,4) 0.0730 0.0904		(6,5) 0.0743 0.0998	
(7,2) 0.0739 0.0998		(7,3) 0.0692 0.0904		(7,4) 0.0692 0.0904		(7,5) 0.0692 0.0904		(7,6) 0.0739 0.0998

Table 2: Structure Selection Tree (Up) The Structural Parameters ( $N, N_c$ ); (Middle) MAE for Boolean Filter; (Bottom) MAE for Stack Filter.