NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Canada

Linearly Separable Stack-Like Architecture for the Design of Weighted Order Statistic Filters with Application in Image Processing

Cristian Emanuel Savin

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfilment of the Requirements

for the Degree of Master of Applied Science at

Concordia University

Montréal, Québec, Canada

September 1993

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

ISBN 0-315-87284-5

Canada

ABSTRACT

Linearly Separable Stack-Like Architecture for the Design of Weighted Order Statistic Filters with Application in Image Processing

Cristian Emanuel Savin

In this thesis, the problem of designing a weighted order statistic (WOS) filter which is approximately optimal in the mean absolute error (MAE) sense, for estimating a signal from the noise-corrupted observation of the same, is considered. A stack filter configuration described by a linearly separable positive Boolean function (LSPBF) and referred to as linearly separable stack (LSS) filter, has been traditionally used for this design. The design of WOS filters in the domain of LSS filter architecture is a constrained design, in the sense that the weights defining the LSPBF can assume only positive or zero values.

This thesis introduces a new approach for the design of WOS filters which are approximately optimal for estimation in the MAE sense, by defining a more general type of filter configuration than that of LSS filters. This new type of architecture is characterized by a linearly separable Boolean function (LSBF), and is designated as linearly separable stack-like (LSSL) filter. In the case of LSSL filters, the weights may assume any value, positive or negative. It is shown that LSSL filters satisfy a new property referred to as generalized stacking property. It is established that due to this property, the fundamentals of the optimality theory that has been developed for the class of stack filters, remain valid in the framework of LSSL filters as well. It is demonstrated that in the multilevel signal domain, an LSSL filter architecture

performs the operation of WOS filtering.

An adaptive algorithm for the design of WOS filters in the domain of LSSL filter architecture is derived. Since the filter weights in this architecture can assume any real values, the algorithm is less constrained than that in the case of LSS filter architecture. Consequently, the proposed design is expected to give better results in the sense of mean absolute error in signal estimation problems. An implementation of the proposed design algorithm is constructed by using a binary-level LMS algorithm.

The proposed design and implementation is applied to the problem of restoring images corrupted with impulsive noise. Simulation results show that the WOS filters designed with the new method provide better results compared with those obtained by using the LSS filter architecture.

ACKNOWLEDGEMENTS

I would like to express my profound gratitude to my supervisors, Dr. M. O. Ahmad and Dr. M. N. S. Swamy, for providing the guidance and support that made this work possible. I am grateful for their extremely careful, thorough, and critical review of my thesis, and for their inspiration throughout the course of this work. I would also like to thank the members of my examination committee, Dr. E. I. Plotkin, Dr. T. Fancott, and Dr. A. K. Elhakeem, for their helpful comments and suggestions.

I would like to express my gratitude to my mother and to my late father, for their love and sacrifices that they made in shaping up my life. I am also deeply grateful to my wife Daniela for her unselfish concern and understanding.

Many thanks are due to my close relatives and friends, especially to Edith and Dorel Feldman, and to Sonia and Hugo Rosman, for their help, advice, and encouragement.

This work was supported partially by the MICRONET, a National Network of Centers of Excellence, and by a Concordia University Graduate Fellowship.

Dedicated to

the memory of

my father

TABLE OF CONTENTS

List of Figures

List of Abbreviations and Symbols

FIR	Finite impulse response
$\mathbf{gr}_\ell(n)$	Binary input sequence appearing at instant n in the filter's window at level ℓ, in the generalized threshold decomposition architecture
L	Number of grey levels of a signal
LMAE	Least mean absolute error
LSBF	Linearly separable Boolean function
LSPBF	Linearly separable positive Boolean function
LSS	Linearly separable stack
LSSL	Linearly separable stack-like
M	Window size of a 1-D filter
MAE	Mean absolute error
MMAE	Minimum mean absolute error
MSE	Mean square error
PBF	Positive Boolean function
$\mathbf{r}_\ell(n)$	Binary input sequence appearing at instant n in the filter's window at level ℓ, in the threshold decomposition architecture
$\mathbf{R}_\mathrm{M}(n)$	Multilevel input sequence appearing at instant n in the filter's window of size M
$\mathcal{S}_f$	Stack filter operator characterized by a positive Boolean function f
$\mathcal{S}\ell_f$	Stack-like filter operator characterized by a linearly separable Boolean function f
WOS	Weighted order statistic

Chapter 1

Introduction

1.1 General

Linear filters have long been considered as the primary tools for signal and image processing. They are easy to implement and analyze, and they are well-suited for the design under the mathematically tractable mean square error criterion. However, this design can be used only when the input signal is corrupted with additive Gaussian noise. It has been experimentally observed that even a small deviation from the Gaussian assumption may sometimes lead to severe deteriorations in the performance of linear filters [GW81].

Currently, in view of the increasing technological demand, better and more specialized tools are needed, for processing signals which are severely corrupted with different types of non-Gaussian noise, such as speckle noise and salt and pepper noise. It has been shown that in this type of applications, nonlinear filters constitute a much better alternative than the linear ones [GW81]. Specifically, when the input signal is corrupted by speckle noise or by salt and pepper noise, rank-order based filters have proved to be especially attractive [GCG92]. However, there are several classes of rank-order based filters, and the task of choosing the right one is itself a challenge.

Two main types of rank-order based filters have been traditionally given a special attention. The first type includes filters in which rank order operations are combined with linear operations in some fashion. It has been shown in [Gab89] and [GCG92], that this type of rank-order based filters has a major drawback, in that the optimization over such classes of filters is mainly reduced to optimizing the linear part of the filter. The class of stack filters is the second major type of rank-order based filters. They perform nonlinear rank-order based operations, and have been introduced in an effort to correct the drawback of the first type of filters [WCG86].

In 1985, a very powerful theoretical tool for analyzing rank order filters was developed by Fitch et al [FCG85]. They showed that these filters possess a limited superposition property, which states that the rank order filtering of an arbitrary multilevel sequence is equivalent to decomposing the signal into binary sequences by thresholding, filtering each binary sequence by a binary rank order filter, and then reversing the decomposition process. This equivalence, reduces the analysis of multilevel sequences to that of binary sequences. The configuration in which a multilevel sequence is reduced to binary sequences is called threshold decomposition architecture.

In addition to the above mentioned limited superposition property of rank order filters in the threshold decomposition architecture, another property, called the stacking property, was identified by Wendt et al [WCG86]. The stacking property is an ordering property which referes to the outputs of the binary rank order filters.

The definition of stack filters was given by abstracting the properties of rank order filters in the threshold decomposition configuration [WCG86]. Specifically, in a stack filter architecture, the binary rank-order based operator is characterized by an arbitrary positive Boolean function (PBF). An optimality theory for the design of stack filters under the mean absolute error (MAE) criterion was later developed by Coyle and Lin [CL88]. Based on this theory, it is possible to determine the stack filter which minimizes the MAE between its output and a desired signal, given

a noise-corrupted observation of the latter. This optimality theory for nonlinear rank-order based filters is as analytically tractable as the theory of optimal linear filtering.

It has been shown that an optimal stack filter can be determined by using a linear program (LP) [CL88]. The complexity of this LP increases exponentially with the size of the filter window. For instance, even for a relatively small window size of 4×4, the required LP contains over one million variables and constraints. Since all the methods which have been developed for the design of stack filters are LP-based optimizations, they all inherit the drawback of being computationally very expensive [GC90], [LSC90], [GC91], [ZGN91].

In order to overcome the difficulties related to the computational complexity in the design of stack filters, a new approach has been initiated in the work of Coyle and Gallagher [CG89] and Harja et al [HAN91]. The main idea of this approach is to substantially reduce the computational burden at the expense of considering only some specific classes of stack filters, that have known practical significance in signal processing. Such an example is the class of stack filters which are characterized by linearly separable PBFs (LSPBFs). These filters, called linearly separable stack (LSS) filters, perform the nonlinear functions of weighted order statistic (WOS) filtering [HAN91]. The WOS filters have been successfully used in a wide variety of signal and image processing applications. Well-known filters like the median, the weighted median, and the rank order operators, are special cases of WOS filters [HAN91]. In the design of WOS filters, the number of optimization variables is equal to the window size of the filter.

1.2 Existing Techniques for the Design of WOS Filters Using LSS Architectures

At present, because of the difficulties involved in imposing the constraint of linear separability for the PBFs, there does not exist a closed form solution for the design of WOS filters by using LSS architectures. Thus, the current practice of designing such filters is based on adaptive techniques [YAN92], [AHL92], [YAN93].

Based on the work of Harja et al [HAN91] and Coyle and Gallagher [CG89], a heuristic configuration of stack filters in which the PBF in the threshold decomposition architecture is replaced by a neural network, was proposed for the design of WOS filters [YAN92], [AHL92]. With the help of this configuration, the design is performed by using a constrained LMS algorithm for adjusting the weights of the neural network. These weights are allowed to assume only positive or zero values, such that the neuron performs the function of an LSPBF. It has been shown that this heuristic approach gives satisfactory experimental results. Recently, a theoretical formulation for the adaptive design of LSS filters has been developed [YAN93], by using a procedure that is similar to Widrow's derivation of the LMS algorithm [WS85]. This design involves a least mean absolute error (LMAE) algorithm. In the adaptive LMAE design of LSS filters, the weights which become negative during the training procedure are reset to zero. This is a constrained optimization, in the sense that the weights can assume either positive or zero values, but they are not allowed to assume negative values. However, there could be situations in which the mean absolute error could be further reduced if the weights were allowed to assume any value, positive or negative.

1.3 Scope and Organization of the Thesis

In this thesis, a new approach for the design of WOS filters which are approximately optimal for estimation in the MAE sense is proposed, by defining a more general type of filter configuration than that of LSS filters. This new type of architecture is characterized by a linearly separable Boolean function (LSBF), and it is referred to as linearly separable stack-like (LSSL) filter. In the case of LSSL filters, the weights may assume any value, positive or negative. It is shown that the fundamentals of the optimality theory which has been developed in [CL88] for the class of stack filters, remain valid in the framework of LSSL filters as well.

An adaptive algorithm for the design of WOS filters in the domain of LSSL architecture is derived. Since the filter weights in this architecture are not restricted and can assume any real values, the algorithm is less constrained than that in the case of LSS filter architecture. Consequently, the proposed design is expected to give better results in the sense of mean absolute error in signal estimation problems. An implementation of the proposed design algorithm is constructed by using a binary-level LMS algorithm.

The thesis is organized as follows. In Chapter 2, a review of the stack filter theory and the relevant background material is presented. The emphasis is placed on the problem of designing stack filters that are optimal for estimation under the MAE criterion. A new and direct approach for the formulation of the linear program which gives the optimal filter is proposed in this chapter. In Chapter 3, the LSSL filter architecture is defined based on introducing a new concept of generalized threshold decomposition of an L-level signal in a window of size M. It is demonstrated that in multilevel signal domain, an LSSL filter architecture performs the operation of WOS filtering. It is also shown that, in a stack-like filter architecture, although the binary input signals do not satisfy the stacking property in the strict sense, the binary output signals do. Based on this property, in Chapter 4, the fundamentals of the optimality theory developed by Coyle and Lin [CL88] are extended to the class of

stack-like filters. An adaptive algorithm for the design of WOS filters in the domain of LSSL architecture is derived, and an implementation of this design by using a binary-level LMS algorithm is proposed. The proposed design and implementation is applied to the problem of restoring images corrupted by impulsive noise. Finally, Chapter 5 summarizes the work of this investigation.

Chapter 2

Stack Filter Theory

2.1 Introduction

A stack filter is a sliding window nonlinear filter whose output at each window position is the result of a superposition of the outputs of a stack of positive Boolean functions operating on thresholded versions of the samples appearing in the filter's window. It will be shown that, for instance, instead of performing a direct median filtering with a window size M = 3 to a multilevel 1-D sequence, it is possible to use a stack architecture and apply the binary function

$$f(x_0, x_1, x_2) = x_0x_1 + x_1x_2 + x_2x_1 \tag{2.1}$$

on each threshold level of the input sequence. The function f given by (2.1) is a positive Boolean function (PBF), which refers to the fact that the expression $x_0x_1 + x_1x_2 + x_2x_1$ contains no complements of the input variables (i.e., $\overline{x}_0$, $\overline{x}_1$, or $\overline{x}_2$).

In what follows, the main concepts that are used in stack filter theory are introduced by considering only the case of 1-D signals, as their 2-D extensions are straightforward. We begin with the definition of a positive Boolean function [Mur71], and then introduce the concept of threshold decomposition of multilevel signals

[FCG84], [FCG85] and the stacking property of binary signals [FCG84], [FCG85], [WCG86]. The stacking property of Boolean functions, which is essential to the understanding of stack filtering, is described in terms of the stacking property of binary sequences. Having introduced the above mentioned concepts, the architecture of a stack filter is described, in which the multilevel input signal is first decomposed into a set of binary signals by threshold decomposition, then a positive Boolean function is performed on each of these threshold signals, and finally, all the binary results, which have the stacking property, are added together yielding the multilevel output.

The theory of optimal stack filtering which is based on the use of the minimum mean absolute error (MMAE) criterion is also introduced in this chapter. The formulation of the optimal filtering problem as an integer linear program is presented [CL88], [LC90], and the adaptive approach in optimal stack filtering [LSC90] is briefly reviewed.

2.2 Definitions and Properties

2.2.1 Positive Boolean Functions

In the following, using the terminology related to switching functions as proposed by Muroga [Mur71], a brief review of some basic definitions of Boolean algebra is given. A binary function is symbolically denoted as $f(\mathbf{x})$, using a vector notation $\mathbf{x}$ for the set of input variables, $(x_0, x_1, \ldots, x_n)$.

Definition 2.1 *A literal is defined as a binary variable or its complement. A conjunction (i.e., a logical product) of literals in which a literal for each variable appears at most once is called a term or a product. In a special case, a conjunction may comprise a single literal. Similarly, a disjunction (i.e., a logical sum or alternation) of literals in which a literal for each variable appears at most once (including the case*

of a single literal) is called an alterm. A disjunction of terms is called a disjunctive form or normal form. A conjunction of alterms is called a conjunctive form.

For example, x_1 and $\overline{x}_1$ are the literals of the variable x_1. Both $x_1x_2 + x_1 + x_2$ and $x_1 + \overline{x}_1x_2$ are disjunctive forms which are equivalent to the Boolean function $x_1 + x_2$. However, the expression $x_1 + x_2(\overline{x}_1 + \overline{x}_2)$ is not a disjunctive form, although it is also equivalent to the same function.

Definition 2.2 *If there exists a disjunctive form for a Boolean function $f(\mathbf{x})$, such that the variable $\overline{x}_j$ (x_j) does not appear in any term of this form, then f is said to be positive (negative) in the variable x_j.*

Definition 2.3 *If a Boolean function $f(\mathbf{x})$ (or a disjunctive form of f) is either positive or negative in a variable x_j, then f is said to be unate in the variable x_j.*

The concepts of unate functions, positive Boolean functions, and negative Boolean functions as defined in [Mur71] are given next.

Definition 2.4 *If a Boolean function $f(\mathbf{x})$ is unate in all its variables, then f is said to be a unate function; a Boolean function which is positive (negative) in all its variables is called a positive (negative) Boolean function.*

For example, the function $x_0 + \overline{x}_1x_2$ is a unate function and the function $x_0 + x_1x_2$ is a positive function. We notice that the function $f = x_0 + \overline{x}_0x_1$ is a positive function because there exists the disjunctive form $x_0 + x_1$, equivalent to f, which is free of the complemented variables $\overline{x}_0$ and $\overline{x}_1$, although the particular disjunctive form $x_0 + \overline{x}_0x_1$ itself is not positive in x_0 [1]. Function $x_0x_1 + \overline{x}_1\overline{x}_2$ is positive in x_0, negative in x_2, and this disjunctive form is neither positive nor negative in x_1 (i.e., it is not unate in x_1).

[1] Note that there is a difference between a disjunctive form of a function being positive and the function itself being positive in a particular variable.

In order to introduce a theorem given by Quine [Qui53], which suggests a method to check whether a Boolean function $f(\mathbf{x})$ is unate or not, we recall the concept of minimal sum-of-product form of a binary function as given in [Mur71].

Definition 2.5 *Let $f(\mathbf{x})$ and $g(\mathbf{x})$ be two Boolean functions. If for every $\mathbf{b}_\alpha$ for which $f(\mathbf{b}_\alpha) = 1$, we also have $g(\mathbf{b}_\alpha) = 1$, we write*

$$f(\mathbf{x}) \subseteq g(\mathbf{x}), \tag{2.2}$$

and we say that f implies g. If, in addition, there exists at least one $\mathbf{b}_\beta$ for which $f(\mathbf{b}_\beta) = 0$ and $g(\mathbf{b}_\beta) = 1$, we write

$$f(\mathbf{x}) \subset g(\mathbf{x}), \tag{2.3}$$

and say that f strictly implies g.

Definition 2.6 *If an implication relation holds between two binary functions f and g, i.e., either $f \subseteq g$ or $g \subseteq f$, then f and g are said to be implication comparable; otherwise, they are implication incomparable.*

An example of implication comparable functions is given by $f = x_1 + x_2x_3$ and $g = x_1 + x_2$, i.e., $x_1 + x_2x_3 \subset x_1 + x_2$.

Definition 2.7 *An implicant of a Boolean function f is a term [2] that implies f.*

For instance, x_0, x_0x_1, $x_0\overline{x}_1$, x_0x_3, x_1x_2 are examples of implicants of the function $x_0 + x_1x_2$.

Definition 2.8 *A term* P *is said to subsume another term* Q *if all literals of* Q *are literals of* P *as well.*

For example, the term $x_0x_1\overline{x}_2$ subsumes the term $x_1\overline{x}_2$.

[2] Note that a term as well as an alterm are also Boolean functions.

Definition 2.9 *A prime implicant of a binary function f is defined as an implicant of f such that no term subsumed by this implicant can be an implicant of f.*

Definition 2.10 *A minimal sum-of-product form of a Boolean function f is a disjunction of prime implicants such that removal of any of them makes the remaining expression no longer equivalent to the original f.*

The following theorem given by Quine [Qui53] shows that in order to determine whether or not a Boolean function f is unate, one has to find a minimal sum-of-product form of f and check if this form is unate. If this minimal sum-of-product form of f is unate, then f is unate, and we can immediately say in which variables the function is positive and in which ones it is negative.

Theorem 2.1 *A unate function f has a unique minimal sum-of-product form and this form is unate.*

As a consequence of this theorem, a positive Boolean function has a unique minimal sum-of-product form in which there are no complements of the input variables.

2.2.2 Threshold Decomposition of Discrete-Time Signals

Threshold decomposition of a discrete-time signal may be formally defined as follows [Gab89].

Definition 2.11 *Let* $\mathbf{R} = (\mathrm{R}(0), \mathrm{R}(1), \ldots, \mathrm{R}(\mathrm{N}-1))$ *denote an* L*-level discrete-time signal of* N *samples. The threshold decomposition of* $\mathbf{R}$ *is the set of* $(\mathrm{L}-1)$ *binary sequences, called threshold signals,* $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_{\mathrm{L}-1}$*, whose elements are defined by*

$$\mathrm{r}_\ell(j) = \begin{cases} 1 & \text{if} \quad \mathrm{R}(j) \geq \ell \\ 0 & \text{if} \quad \mathrm{R}(j) < \ell \end{cases} \tag{2.4}$$

for $\ell = 1, 2, \ldots, \mathrm{L}-1$*, and* $j = 0, 1, \ldots, \mathrm{N}-1$.

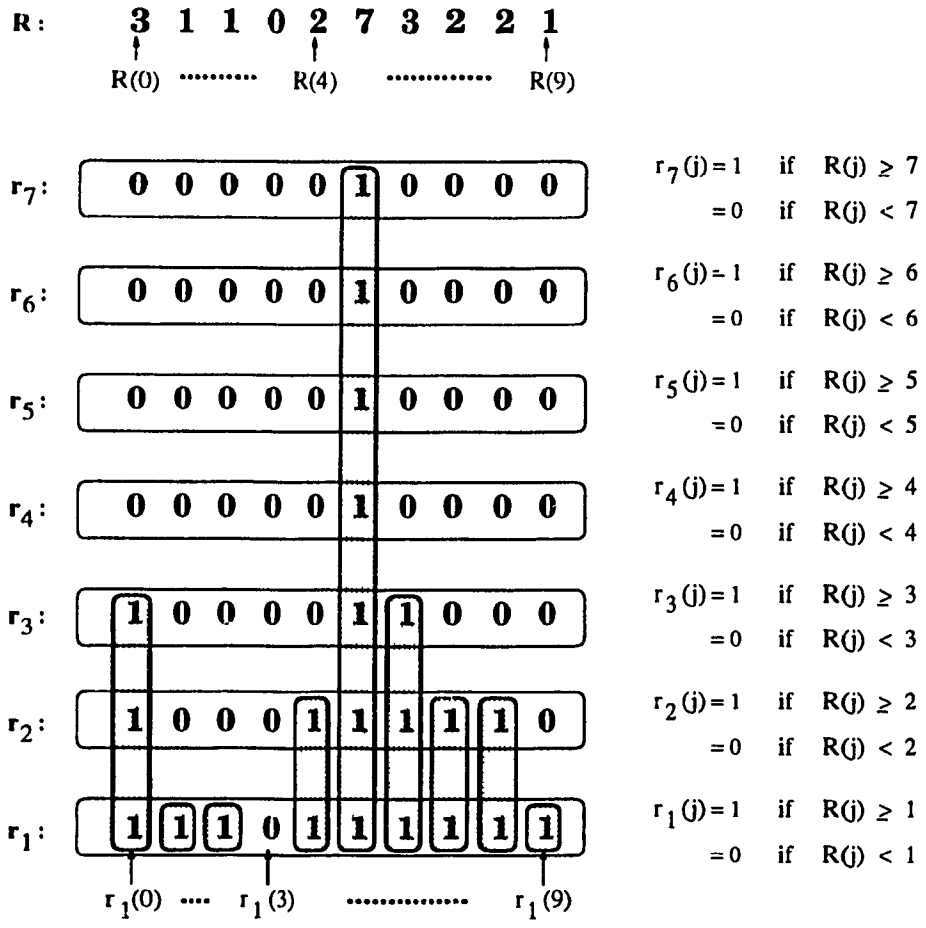


Figure 2.1: An example of threshold decomposition of a discrete-time signal $\mathbf{R}$ with N=10 and L=8.

Based on Definition 2.11, we can express $\mathbf{R}$ and $R(j)$ as follows:

$$\mathbf{R} = \sum_{\ell=1}^{L-1} \mathbf{r}_\ell \ , \tag{2.5}$$

$$R(j) = \sum_{\ell=1}^{L-1} r_\ell(j) \quad \text{for} \quad j = 0, 1, \ldots, N-1 \, . \tag{2.6}$$

Note that the binary signals $\mathbf{r}_\ell$'s are Boolean vectors. An example of threshold decomposition of a multilevel sequence consisting of $N = 10$ samples with $L = 8$ is shown in Figure 2.1.

2.2.3 Stacking Property of a Set of Binary Signals

In order to introduce the stacking property of a set of binary signals (Boolean vectors), each of the same finite length N, we first define their ordering relation [Mur71].

Definition 2.12 *If two Boolean vectors* $\mathbf{a} = (a_0, a_1, \ldots, a_{N-1})$ *and* $\mathbf{b} = (b_0, b_1, \ldots, b_{N-1})$ *satisfy the condition that*

$$a_j \geq b_j \quad \forall\, j = 0, 1, \ldots, N-1\,,$$

then $\mathbf{a}$ *is said to be greater than or equal to* $\mathbf{b}$, *i.e.,*

$$\mathbf{a} \geq \mathbf{b}\,.$$

In particular, if there exists a j *for which* $a_j > b_j$, *and* $a_k \geq b_k$ *for all* $k \neq j$, *then* $\mathbf{a}$ *is said to be greater than* $\mathbf{b}$ *or* $\mathbf{b}$ *smaller than* $\mathbf{a}$, *i.e.,*

$$\mathbf{a} > \mathbf{b}\,.$$

Definition 2.13 *When for two binary vectors,* $\mathbf{c}$ *and* $\mathbf{d}$, *none of the relations* $\mathbf{c} > \mathbf{d}$, $\mathbf{c} < \mathbf{d}$, *or* $\mathbf{c} = \mathbf{d}$ *holds, then* $\mathbf{c}$ *and* $\mathbf{d}$ *are said to be incomparable.*

The stacking property for a set of binary signals can now be stated as follows [Gab89].

Property 2.1 *A set of* $L-1$ *binary signals of length* N, $\{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_{L-1}\}$, *is said to have the stacking property if the following relation holds:*

$$\mathbf{r}_1 \geq \mathbf{r}_2 \geq \ldots \geq \mathbf{r}_{L-1}\,.$$

By the definition of threshold decomposition of an L-level discrete-time signal (see Definition 2.11), the threshold signals $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_{L-1}$ will always satisfy the stacking property. For instance, the binary signals of the example shown in Figure 2.1, obtained as a result of threshold decomposition satisfy the stacking property.

2.2.4 Stacking Property of a Boolean Filter

The stacking property of a Boolean filter is stated in terms of the stacking property of its binary input and output signals as given below [FCG85], [WCG86], [Gab89].

Property 2.2 *An* M*-input Boolean filter (function)* $f(x_0, x_1, \ldots, x_{M-1})$ *is said to satisfy the stacking property if and only if*

$$f(\mathbf{b}_\alpha) \geq f(\mathbf{b}_\beta) \quad \text{whenever} \quad \mathbf{b}_\alpha \geq \mathbf{b}_\beta \ , \qquad \text{for all} \quad \alpha, \beta \in \{1, 2, \ldots, 2^M\} \ .$$

The necessary and sufficient condition for a Boolean function (or filter) to satisfy the stacking property was stated by Gilbert [Gil54], [Mur71].

Theorem 2.2 *A Boolean function satisfies the stacking property if and only if it is a positive Boolean function.*

Stacking property of PBFs is suggestively highlighted by the so-called *Hasse diagram* [She69], which is also referred to as the *stacking diagram* [GC91]. A brief description for the construction of such a diagram is now given.

Let us partition the total number of 2^M possible input binary vectors $\mathbf{b}_\alpha$ ($\alpha = 1, 2, \ldots, 2^M$) to a Boolean filter of size M with respect to the number of 1's in each binary vector, as given below:

$$\mathbf{V}_k = \{\mathbf{b}_\alpha \mid \mathbf{b}_\alpha \cdot \mathbf{1}_{M\times 1} = k\} \quad for \quad k = 0, 1, \ldots, M \ , \tag{2.7}$$

where $\mathbf{1}_{M\times 1}$ is a column vector of size M whose elements are all 1's. The vectors in each group $\mathbf{V}_k$ are incomparable. Each vertex in the stacking diagram represents a binary vector, and the number of vertices in each group $\mathbf{V}_k$ is given by the binomial number $\binom{k}{M}$. The set of all binary vectors is denoted by $\mathbf{V}$:

$$\mathbf{V} = \bigcup_{k=0}^{M} \mathbf{V}_k \ . \tag{2.8}$$

The distance between two vectors $\mathbf{b}_\alpha$ and $\mathbf{b}_\beta$ is defined as

$$d(\mathbf{b}_\alpha, \mathbf{b}_\beta) = |\mathbf{b}_\alpha \cdot \mathbf{1}_{M\times 1} - \mathbf{b}_\beta \cdot \mathbf{1}_{M\times 1}| \ . \tag{2.9}$$

For instance, if $\mathbf{b}_\alpha \in \mathbf{V}_j$ and $\mathbf{b}_\beta \in \mathbf{V}_k$, then $d(\mathbf{b}_\alpha, \mathbf{b}_\beta) = |j - k|$. Therefore, $\mathcal{V} = \{\mathbf{V}_0, \mathbf{V}_1, \ldots, \mathbf{V}_M\}$ is an ordered set of the groups $\mathbf{V}_k$'s of binary vectors, such that the distance between a pair of vectors taken from two consecutive groups of $\mathcal{V}$ is one, and the first group is always $\mathbf{V}_0$. Once the set $\mathcal{V}$ has been determined, the groups $\mathbf{V}_k$'s are stacked on top of each other in the same order as in $\mathcal{V}$, and the vertices of each pair $(\mathbf{b}_\alpha, \mathbf{b}_\beta)$ for which $\mathbf{b}_\alpha$ and $\mathbf{b}_\beta$ are comparable and $d(\mathbf{b}_\alpha, \mathbf{b}_\beta) = 1$ are connected by an edge.

Figure 2.2 illustrates the stacking diagram for the function $f(x_0, x_1, x_2, x_3) = x_1 + x_0x_2x_3$, which is a PBF, and therefore it satisfies the stacking property. For instance, because $f(0100) = 1$, we also have $f(1100) = 1$, $f(0101) = 1$, and $f(0110) = 1$. Analogously, as $f(1001) = 0$, we also have $f(1000) = 0$ and $f(0001) = 0$, since in general, if a binary function f satisfies the stacking property, then for each edge $[(\mathbf{p}, 0, \mathbf{q}), (\mathbf{p}, 1, \mathbf{q})]$ of the stacking diagram of f the following relation holds:

$$f(\mathbf{p}, 1, \mathbf{q}) \geq f(\mathbf{p}, 0, \mathbf{q}) \,. \tag{2.10}$$

Stacking diagram might be formally described as given below [GC91]:

Definition 2.14 *The stacking diagram for a Boolean filter of size* M *is the undirected graph* $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, *where* $\mathbf{V}$ *is the set of all binary vectors of size* M, *and the set of edges,* $\mathbf{E}$, *consists of*

$$\mathbf{E} = \{(\mathbf{b}_\alpha, \mathbf{b}_\beta) \mid \mathbf{b}_\alpha, \mathbf{b}_\beta \in \mathbf{V}, \textit{ are comparable, and } d(\mathbf{b}_\alpha, \mathbf{b}_\beta) = 1\} \,. \tag{2.11}$$

Note that if two Boolean functions f and g are implication comparable (see Definition 2.6), then

$$f(\mathbf{b}_\alpha) \leq g(\mathbf{b}_\beta) \quad \text{whenever} \quad \mathbf{b}_\alpha \leq \mathbf{b}_\beta \,, \qquad \text{for all} \quad \alpha, \beta \in \{1, 2, \ldots, 2^M\} \,.$$

For example, functions f and g for which the stacking diagrams are shown in Figures 2.2 and 2.3, respectively, are implication comparable. Specifically, f strictly implies g, i.e., $f \subset g$.

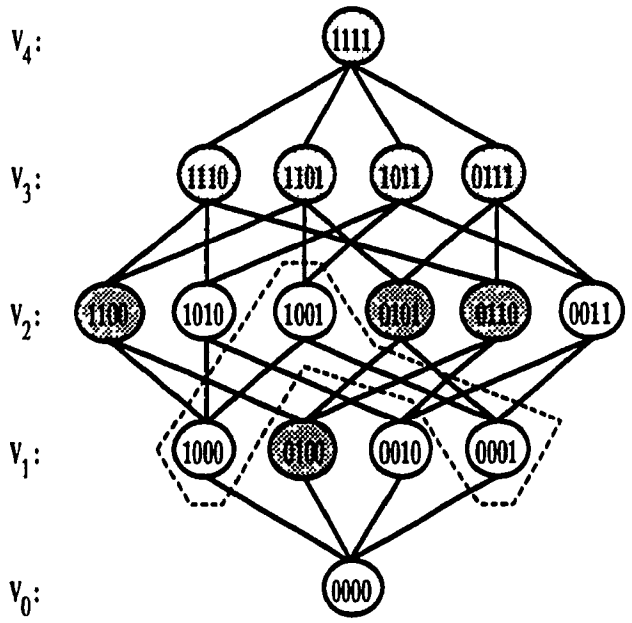


Figure 2.2: Stacking diagram for a Boolean filter $f(x_0, x_1, x_2, x_3) = x_1 + x_0x_2x_3$, where a shaded ellipse surrounding a binary vector $\mathbf{b}_\alpha$ signifies that $f(\mathbf{b}_\alpha) = 1$, and an unshaded ellipse indicates that $f(\mathbf{b}_\alpha) = 0$.

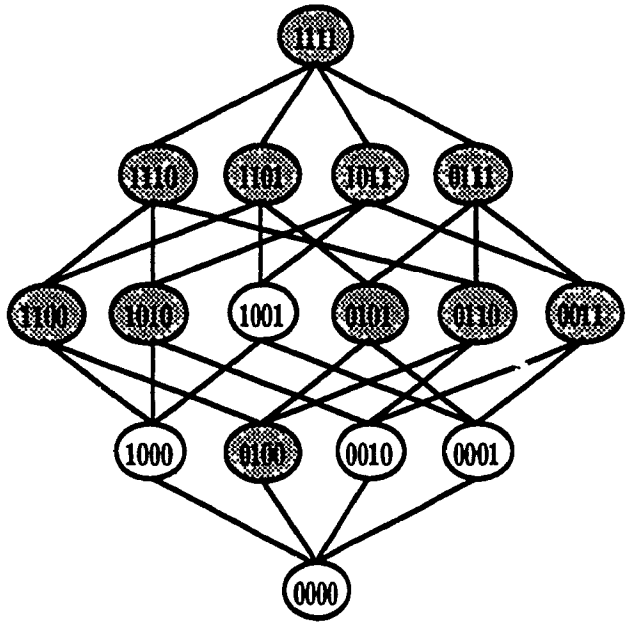


Figure 2.3: Stacking diagram for the Boolean filter $g(x_0, x_1, x_2, x_3) = x_1 + x_0x_2 + x_2x_3$; note that for the function f of Figure 2.2 we have $f \subset g$.

2.2.5 The Definition of a Stack Filter

A stack filter is defined by an architecture in which its multilevel input signal is first decomposed into a set of binary signals by thresholding, then a filtering is performed via a PBF on each of these threshold signals, and finally, all the binary results are added together yielding the filter's multilevel output. As an example, Figure 2.4(b) shows a stack filter architecture that performs the operation of median filtering given in Figure 2.4(a).

As the Boolean filter used at each level ℓ ($\ell = 1, 2, \ldots, \mathrm{L}-1$) of the stack filter architecture is a PBF, the Boolean filter has the stacking property. The binary outputs at each time instant n have a structure in which a stack of 0's is piled on top of a stack of 1's. In other words, the binary output signals also satisfy the stacking property.

The mathematical expression for the output of a stack filter operating on an input process can be derived following the procedure given in [GCG92]. Let $\mathrm{R}(n)$ be the process at the input (the received process) of a stack filter, and assume that $\mathrm{R}(n)$ takes on integer values in $\mathbf{Q} = \{0, 1, 2, \ldots, \mathrm{L}-1\}$. The received process $\mathrm{R}(n)$ is assumed to be a noise corrupted version of some desired process $\mathrm{S}(n)$. A window of size M is slid by increments of one sample across the input process $\mathrm{R}(n)$. At each time instant n, the stack filter $S_f(\cdot)$ maps the samples in the window, which are given by

$$\mathbf{R}_{\mathrm{M}}(n) = \left\{ \mathrm{R}(n-\mathrm{M}+1), \quad \ldots, \quad \mathrm{R}(n) \right\}, \tag{2.12}$$

to some integer $S_f(\mathbf{R}_{\mathrm{M}}(n))$ in $\mathbf{Q}$.[3] Thus, a stack filter is a mapping $S_f(\cdot) : \mathbf{Q}^{\mathrm{M}} \rightarrow \mathbf{Q}$

[3]Note that in image processing applications it is a common practice to consider noncausal windows such as

$$\mathbf{R}'_{\mathrm{M}}(n) = \left\{ \mathrm{R}\left(n - \frac{M+1}{2} \right), \quad \ldots, \quad \mathrm{R}(n), \quad \ldots, \quad \mathrm{R}\left(n + \frac{M+1}{2} \right) \right\},$$

instead of $\mathbf{R}_{\mathrm{M}}(n)$ as defined by (2.12).

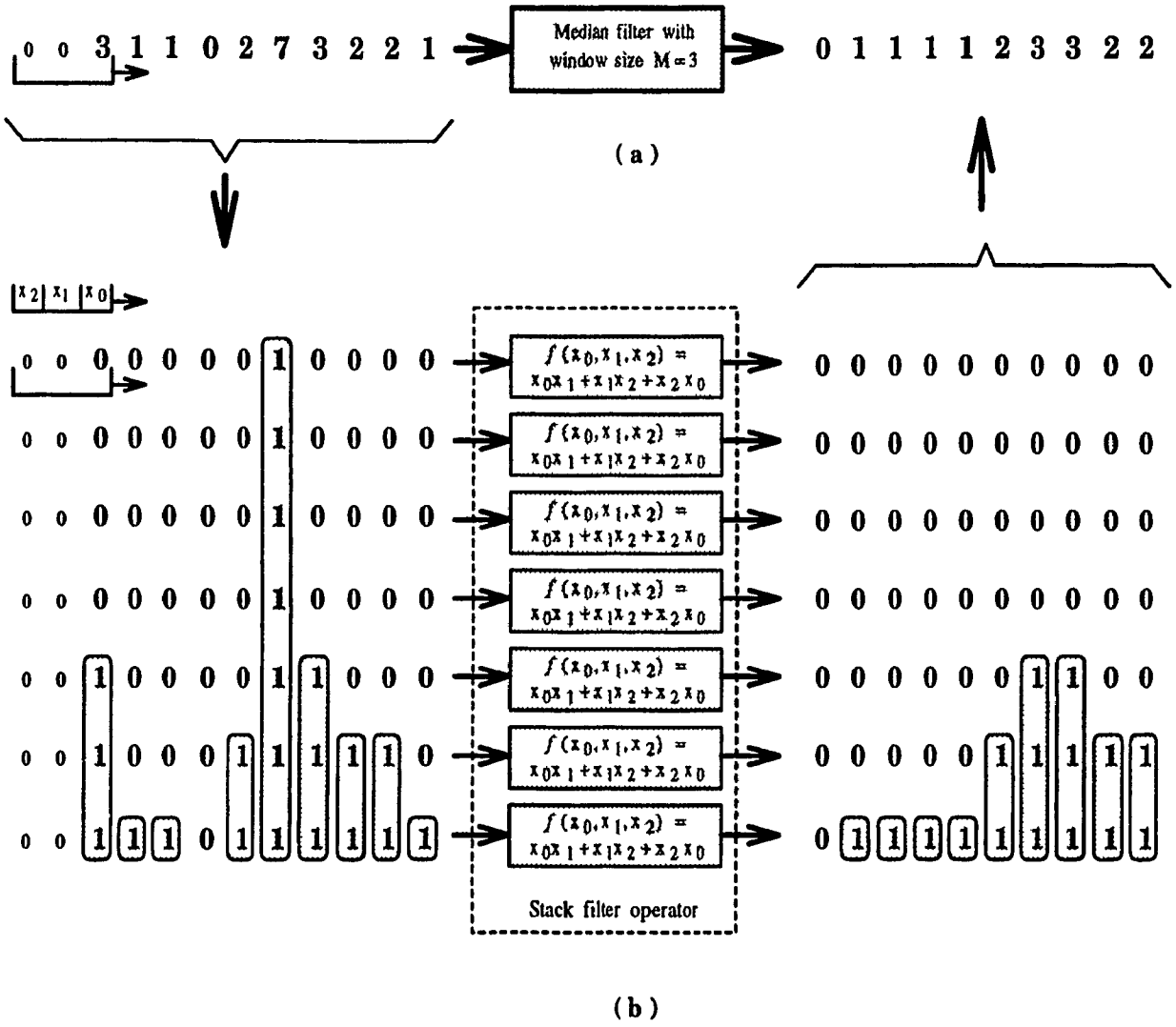


Figure 2.4: An example of a stack filter performing median filtering. (a) A median filter of size M=3 operating on a multilevel input signal with L=8 and N=10. (b) A stack filter architecture that performs the PBF operation corresponding to the median filtering shown in (a).

defined in the framework of the threshold decomposition architecture, i.e.,

$$S_f(\mathbf{R}_M(n)) = \sum_{\ell=1}^{L-1} f(t_\ell(\mathbf{R}_M(n))) , \tag{2.13}$$

where $f(\cdot)$ denotes the Boolean function used at each level of the architecture, which is assumed to posses the stacking property, and

$$\mathbf{t}_\ell(\mathbf{R}_M(n)) = \Big(\; t_\ell(R(n-M+1)), \quad \ldots, \quad t_\ell(R(n)) \; \Big) , \tag{2.14}$$

with

$$t_\ell(R(j)) = \begin{cases} 1 & \text{if } R(j) \geq \ell \\ 0 & \text{if } R(j) < \ell \end{cases} \qquad \text{for all} \quad j = n-M+1, n-M+2, \ldots, n . \tag{2.15}$$

The stack filter architecture exhibits a so-called *weak superposition property* [FCG84], [FCG85], since when the input multilevel signal is decomposed by thresholding, we have

$$S_f(\mathbf{R}_M(n)) = S_f(\sum_{\ell=1}^{L-1} \mathbf{r}_{\ell,M}(n)) = \sum_{\ell=1}^{L-1} f(\mathbf{r}_{\ell,M}(n)) , \tag{2.16}$$

where $\mathbf{R}_M(n)$ denotes the multilevel input sequence which appears in the filter's window of size M at the time instant n and $\mathbf{r}_{\ell,M}(n)$ represents the corresponding set of threshold signals ($\mathbf{r}_{\ell,M}(n) \equiv \mathbf{t}_\ell(\mathbf{R}_M(n))$). For convenience, in what follows we will not use the subscript M for the threshold signals of $\mathbf{R}_M(n)$ anymore. Therefore, we denote

$$\mathbf{r}_\ell(n) = \mathbf{t}_\ell(\mathbf{R}_M(n)) . \tag{2.17}$$

A stack filter is completely characterized by a positive Boolean operator which is used at each level of the architecture. This is the reason why this Boolean operator is sometimes referred to as a Boolean stack filter. Note that, in general, a stack filter architecture can be allowed to consist of different Boolean filters operating at different levels of the structure, with the restriction that the Boolean stack filters should satisfy the implication relation (see Definition 2.5) in an appropriate

sequence. If f_ℓ denotes the binary stack filter which is used at level ℓ of a stack filter architecture, then the following implication relation should hold

$$f_{L-1} \subseteq f_{L-2} \subseteq \cdots \subseteq f_\ell \subseteq \cdots \subseteq f_1 \,. \tag{2.18}$$

However, in this thesis we are concerned only with those stack filters that are characterized by a single PBF.

As shown in [GCG92], a stack filter architecture characterized by a single positive Boolean function f inherits some very attractive features due to the stacking property of f. They are listed below.

1. This architecture allows efficient VLSI implementations of rank-order filters directly in the threshold decomposition architecture [Fit87]. A rank order operation applied to a binary signal does not require sorting, as all possible ranks can be determined from the sum of the input bits, with the desired rank being selected by comparison of the sum with a threshold.

2. It makes possible to perform the summation of the outputs of the Boolean functions by using a binary-tree search technique for the threshold level of the highest 1 in the stack.

3. Due to the stacking property, the level-crossing decisions made by the binary filters at different levels are constrained to be consistent with each other, i.e., the filter at level ℓ will not decide that the signal is less than ℓ, by putting out a 0, if the filter at level $\ell + 1$ decides that the signal is greater than or equal to $\ell + 1$. Therefore, the stacking property of a stack filter architecture can be interpreted as a consistency condition in the context of estimation [CL88], [LSC90].

4. Any operation from $\mathbf{Q}^M$ to $\mathbf{Q}$ (where $\mathbf{Q}$ denotes the set of multilevel input values) has a correspondence in the threshold decomposition architecture with

the Boolean operators at all the threshold levels satisfying the implication property relative to each other (see (2.18)) [LC90].

2.3 Stack Filters and MMAE Estimation

The main interest in stack filter theory lies in the existence of an analytical technique for finding the stack filter which is optimal for estimation under the mean absolute error (MAE) criterion [CL88]. The attractive feature of using MAE criterion for the design of a stack filter is that the stacking property of PBFs allows the decomposition of the estimation error of the filter into the sum of the decision errors incurred by the Boolean operators at each level of the stack filter architecture.

2.3.1 MAE Criterion and Stacking Property of PBFs

The estimation problem for the class of stack filters can be stated using Figure 2.5. The process $\mathrm{R}(n)$ which is received at the input of a stack filter is assumed to be a corrupted version of some desired process $\mathrm{S}(n)$ through an operation $g(\cdot,\cdot)$. The corruption may be caused either by a noise process $\mathrm{N}(n)$ or by some intentional operation, such as a modulation scheme. At each time instant n, the stack filter output is an estimate, called $\hat{\mathrm{S}}(n)$, of the desired process $\mathrm{S}(n)$. This estimate is based on the sequence $\mathbf{R}_{\mathrm{M}}(n)$ observed in the input window of the stack filter

$$\hat{\mathrm{S}}(n) = \mathcal{S}_f(\mathbf{R}_{\mathrm{M}}(n))\,. \tag{2.19}$$

The mean absolute error between the desired signal, $\mathrm{S}(n)$, and the estimated signal, $\hat{\mathrm{S}}(n)$, at the time instant n, is defined to be the cost of using the stack filter $\mathcal{S}_f$ at time n, and it can be expressed as

$$\mathrm{C}_n(\mathcal{S}_f) = E\left[\,\left|\, \mathrm{S}(n) - \mathcal{S}_f(\mathbf{R}_{\mathrm{M}}(n)) \,\right|\,\right]\,. \tag{2.20}$$

If we make the assumption that the signal and noise processes are jointly stationary, then the cost function $\mathrm{C}_n(\mathcal{S}_f)$ has the same value at all time instants n. Therefore,

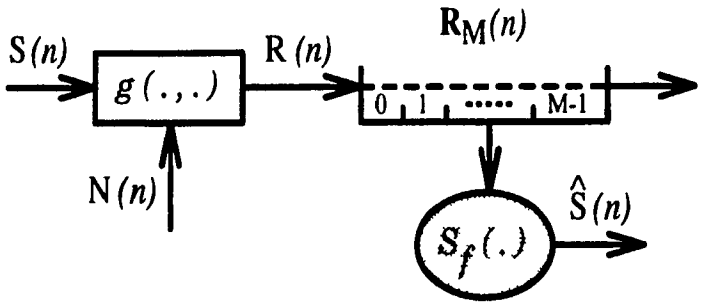


Figure 2.5: The structure of the estimation problem for the class of stack filters.

in the subsequent discussion, the subscript n is not used for the cost function. The optimal stack filter for estimation under mean absolute error criterion is the filter for which $\mathrm{C}(\mathcal{S}_f)$ is minimum.

The rationale of choosing the MAE criterion for the design of a stack filter, is that it allows to express the estimation error of a stack filter as the sum of the decision errors incurred by the Boolean filter at each level of the stack filter architecture. By substituting in (2.20) for $\mathcal{S}_f(\mathbf{R}_\mathrm{M}(n))$ as given by (2.13), and decomposing the desired signal $\mathrm{S}(n)$ by thresholding (see Eq. (2.5)), we have

$$\begin{aligned}
\mathrm{C}(\mathcal{S}_f) &= E\left[\,\left|\, \mathrm{S}(n) - \mathcal{S}_f(\mathbf{R}_\mathrm{M}(n)) \,\right|\,\right] \\
&= E\left[\,\left|\, \sum_{\ell=1}^{\mathrm{L}-1} \mathrm{s}_\ell(n) - \sum_{\ell=1}^{\mathrm{L}-1} f(\mathrm{t}_\ell(\mathbf{R}_\mathrm{M}(n))) \,\right|\,\right] \\
&= E\left[\,\left|\, \sum_{\ell=1}^{\mathrm{L}-1} \left[\, \mathrm{s}_\ell(n) - f(\mathrm{t}_\ell(\mathbf{R}_\mathrm{M}(n))) \,\right] \,\right|\,\right] .
\end{aligned} \tag{2.21}$$

Due to the stacking property of the binary operator f, all the nonzero terms of the sum in (2.21) have the same sign, and therefore, the operations of summation and taking the absolute value may be interchanged as

$$\mathrm{C}(\mathcal{S}_f) = E\left[\, \sum_{\ell=1}^{\mathrm{L}-1} \left|\, \mathrm{s}_\ell(n) - f(\mathrm{t}_\ell(\mathbf{R}_\mathrm{M}(n))) \,\right| \,\right] . \tag{2.22}$$

The operations of finding the expected value and summation in (2.22) can also be interchanged, since the mean value of a sum of random variables equals the sum of

their mean values [Pee80]. Thus, (2.22) becomes

$$\begin{aligned} C(\mathcal{S}_f) &= \sum_{\ell=1}^{L-1} \left[E\left| s_\ell(n) - f(\mathbf{t}_\ell(\mathbf{R}_M(n))) \right| \right] \\ &= \sum_{\ell=1}^{L-1} \left[E\left| s_\ell(n) - f(\mathbf{r}_\ell(n)) \right| \right] , \end{aligned} \tag{2.23}$$

where we have used the notation given in (2.17), that is $\mathbf{r}_\ell(n) = \mathbf{t}_\ell(\mathbf{R}_M(n))$.

The ℓth term of the summation in (2.23) is the mean absolute error incurred at level ℓ, $E[\,|\,e_\ell(n)\,|\,]$, when the Boolean operator f is used at that level. Therefore, (2.23) can be written as

$$C(\mathcal{S}_f) = \sum_{\ell=1}^{L-1} E[\,|\,e_\ell(n)\,|\,] , \tag{2.24}$$

where $e_\ell(n)$ is given by

$$e_\ell(n) = s_\ell(n) - f(\mathbf{t}_\ell(\mathbf{R}_M(n))) . \tag{2.25}$$

Figure 2.6 illustrates the errors incurred in signal estimation using a stack filter architecture, for two possible cases, $\hat{S}(n) \leq S(n)$ and $\hat{S}(n) > S(n)$. In conclusion, the mean absolute error estimation using a stack filter architecture is equivalent to a massively parallel decision making process, in which, at each level ℓ, it is determined whether or not the estimated signal is less than ℓ, when these decisions are constrained to be consistent with each other as required by the stacking property [CL88].

2.3.2 The Optimization Problem to Determine an MMAE Stack Filter

As shown in Section 2.3.1, when f is restricted to be a PBF (i.e., when f satisfies the stacking property), the cost function $C(\mathcal{S}_f)$ defined by (2.21) can be expressed as in (2.23). The problem of designing a stack filter which is optimal for estimation

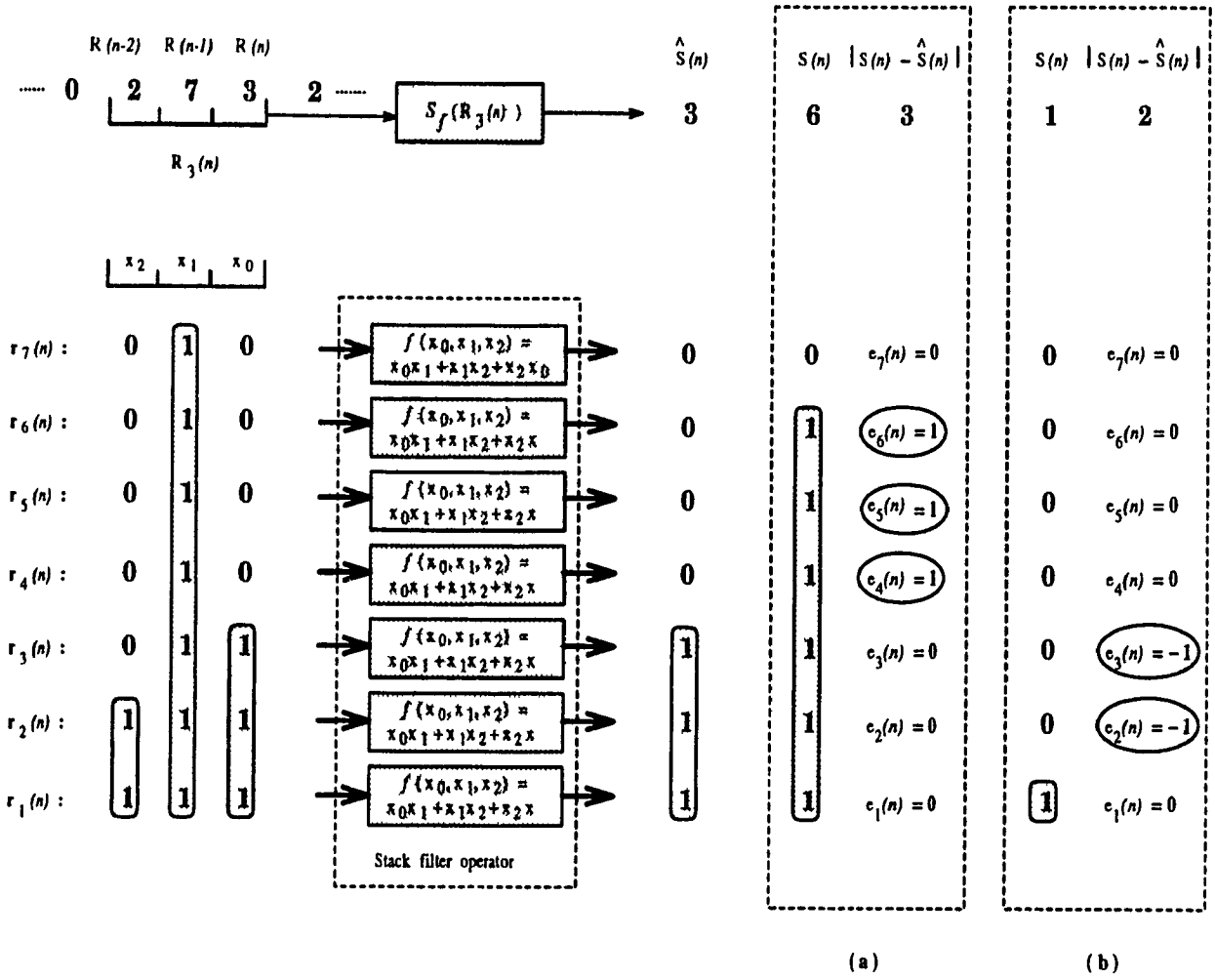


Figure 2.6: An illustration of the errors incurred between the desired and estimated samples in a stack filter architecture, for the two possible cases ($\hat{S}(n) \leq S(n)$ and $\hat{S}(n) > S(n)$). (**a**) When the desired sample is greater than the estimated one, the binary errors at the various threshold levels can assume values of only 0 or 1. (**b**) When the desired sample is less than the estimated one, they can be only 0 or -1.

under the MAE criterion is concerned with minimizing this cost function, and this problem can be formally stated as follows.

Optimization Problem

Determine a Boolean filter $\hat{f} \in \mathcal{F}_{\mathrm{M}}$, *where* $\mathcal{F}_{\mathrm{M}}$ *is the set of all Boolean filters of window size* M, *which achieves*

$$\mathrm{C}(\mathcal{S}_{\hat{f}}) = \min_{f \in \mathcal{F}_{\mathrm{M}}} \sum_{\ell=1}^{\mathrm{L}-1} E\left[\,|\, \mathrm{s}_\ell(n) - f(\mathbf{r}_\ell(n)) \,|\,\right] , \tag{2.26}$$

subject to the constraint that f *should satisfy the stacking property.*

Note that the total number of Boolean filters of window size M is $2^{2^{\mathrm{M}}}$, while the total number of PBFs with M variables is known to be approximatively $2^{2^{\mathrm{M}/2}}$ [CL88].

A Boolean filter f of window size M is completely determined by the vector

$$\mathbf{F} = (\, f(\mathbf{b}_1),\, f(\mathbf{b}_2),\, \ldots,\, f(\mathbf{b}_j),\, \ldots,\, f(\mathbf{b}_{2^{\mathrm{M}}}) \,) , \tag{2.27}$$

where $\left\{\mathbf{b}_j | j = 1, 2, \ldots, 2^{\mathrm{M}}\right\}$ is an ordered sequence of all Boolean vectors of size M. The jth entry in (2.27) represents the output of the filter, when the binary vector in the filter's input window is $\mathbf{b}_j$. Therefore, in order to solve the optimization problem as formulated above, we have to first find an explicit expression for $E[\,|\, \mathrm{s}_\ell(n) - f(\mathbf{r}_\ell(n)) \,|\,]$ as a function of the variables $f(\mathbf{b}_j) \in \{0, 1\}$ $(j = 1, 2, \ldots, 2^{\mathrm{M}})$.

Let us consider the experiment of observing different realizations of the desired threshold process at the level ℓ of a stack filter architecture at the time instant n, $\mathrm{s}_\ell(n) \in \{0, 1\}$, and the corresponding realizations of the binary input vector process to the Boolean filter f at level ℓ, $\mathbf{r}_\ell(n) \in \{\mathbf{b}_1, \mathbf{b}_2, \ldots \mathbf{b}_{2^{\mathrm{M}}}\}$. The sample space (i.e., the set of all possible outcomes) of this experiment is given by

$$\{\, (0, \mathbf{b}_1),\, (0, \mathbf{b}_2),\, \ldots,\, (0, \mathbf{b}_{2^{\mathrm{M}}}),\, (1, \mathbf{b}_1),\, (1, \mathbf{b}_2),\, \ldots,\, (1, \mathbf{b}_{2^{\mathrm{M}}}) \,\} , \tag{2.28}$$

where a pair $(0, \mathbf{b}_j)$ denotes the outcome $(\mathrm{s}_\ell(n) = 0, \mathbf{r}_\ell(n) = \mathbf{b}_j)$, while a pair $(1, \mathbf{b}_j)$ corresponds to $(\mathrm{s}_\ell(n) = 1, \mathbf{r}_\ell(n) = \mathbf{b}_j)$. The expected value of an arbitrary discrete random variable Y, defined on a sample space with K possible outcomes, is given by

$$E\,[\,\mathrm{Y}\,] = \sum_{j=1}^{\mathrm{K}} y_j \cdot P(y_j)\,, \tag{2.29}$$

where y_j denotes a particular value of Y, having the probability of occurrence $P(y_j)$ [Pee80]. Therefore, for the expected value of the random variable $|\,e_\ell(n)\,| = |\,\mathrm{s}_\ell(n) - f(\mathbf{r}_\ell(n))\,|$ (which is a real function of the elements in the sample space (2.28)), we have

$$\begin{aligned} E\,[\,|\,e_\ell(n)\,|\,] &= \sum_{j=1}^{2^{\mathrm{M}}} \big(\,|\,0 - f(\mathbf{b}_j)\,| \cdot P\,[\mathrm{s}_\ell(n) = 0, \mathbf{r}_\ell(n) = \mathbf{b}_j] \\ &\quad + |\,1 - f(\mathbf{b}_j)\,| \cdot P\,[\mathrm{s}_\ell(n) = 1, \mathbf{r}_\ell(n) = \mathbf{b}_j]\,\big) \\ &= \sum_{j=1}^{2^{\mathrm{M}}} \Big(P(0, \mathbf{b}_j|\ell) \cdot f(\mathbf{b}_j) + P(1, \mathbf{b}_j|\ell) \cdot \bar{f}(\mathbf{b}_j)\Big) \\ &= \sum_{j=1}^{2^{\mathrm{M}}} [(P(0, \mathbf{b}_j|\ell) - P(1, \mathbf{b}_j|\ell)) \cdot f(\mathbf{b}_j) + P(1, \mathbf{b}_j|\ell)]\,, \end{aligned} \tag{2.30}$$

where the notations $P(0, \mathbf{b}_j|\ell)$ and $P(1, \mathbf{b}_j|\ell)$ imply, respectively, $P\,[\mathrm{s}_\ell(n) = 0, \mathbf{r}_\ell(n) = \mathbf{b}_j]$ and $P\,[\mathrm{s}_\ell(n) = 1, \mathbf{r}_\ell(n) = \mathbf{b}_j]$. Therefore, the decision error $E\,[\,|\,e_\ell(n)\,|\,]$ at each level ℓ of a stack filter architecture is a linear function of the variables $f(\mathbf{b}_j)$ $(j = 1, 2, \ldots, 2^{\mathrm{M}})$. Using (2.30), the cost function $\mathrm{C}(\mathcal{S}_f)$ in (2.23) becomes

$$\begin{aligned} \mathrm{C}(\mathcal{S}_f) &= \sum_{j=1}^{2^{\mathrm{M}}} \left[f(\mathbf{b}_j) \cdot \left(\sum_{\ell=1}^{\mathrm{L}-1} P(0, \mathbf{b}_j|\ell) \right) + \bar{f}(\mathbf{b}_j) \cdot \left(\sum_{\ell=1}^{\mathrm{L}-1} P(1, \mathbf{b}_j|\ell) \right) \right] \\ &= \sum_{j=1}^{2^{\mathrm{M}}} [\alpha_j \cdot f(\mathbf{b}_j) + \beta_j \cdot \bar{f}(\mathbf{b}_j)]\,, \end{aligned} \tag{2.31}$$

where $\alpha_j = \sum_{\ell=1}^{\mathrm{L}-1} P(0, \mathbf{b}_j|\ell)$ and $\beta_j = \sum_{\ell=1}^{\mathrm{L}-1} P(1, \mathbf{b}_j|\ell)$. Denoting $\gamma_j = \alpha_j - \beta_j$, (2.31) can alternately be written as

$$\mathrm{C}(\mathcal{S}_f) = \sum_{j=1}^{2^{\mathrm{M}}} [\gamma_j \cdot f(\mathbf{b}_j) + \beta_j]\,. \tag{2.32}$$

Finally, the optimization problem can be stated as:

$$\begin{cases} \textit{Determine a Boolean filter } \tilde{f} \textit{ of window size } \mathrm{M}, \textit{ which achieves} \\ \qquad \mathrm{C}(\mathcal{S}_{\tilde{f}}) = \min_{f \in \mathcal{F}_{\mathrm{M}}} \sum_{j=1}^{2^{\mathrm{M}}} [\gamma_j \cdot f(\mathbf{b}_j) + \beta_j] \ , \qquad (2.33) \\ \textit{subject to the following constraints in the stacking diagram :} \\ \qquad \textit{for each vertex } \mathbf{b}_j \ : \ f(\mathbf{b}_j) \in \{0,1\} \ , \qquad (2.34) \\ \textit{and for each edge } [(\mathbf{p},0,\mathbf{q}),(\mathbf{p},1,\mathbf{q})] \ : \ f(\mathbf{p},1,\mathbf{q}) \geq f(\mathbf{p},0,\mathbf{q}) \ . \qquad (2.35) \end{cases}$$

The set of inequalities (2.35) imposes the constraint that f should satisfy the stacking property, that is, it is a PBF.

Assuming that the probability models of the signal and noise processes are given, one can determine the joint probabilities $P(0,\mathbf{b}_j|\ell)$ and $P(1,\mathbf{b}_j|\ell)$, and then calculate the constant coefficients α_j's and β_j's. In general, it is a difficult task to determine the joint probabilities $P(0,\mathbf{b}_j|\ell)$ and $P(1,\mathbf{b}_j|\ell)$. However, in [CL88], Coyle and Lin have discussed this problem in the context of the specific case when the signal and noise are independent Markov chains, and they have developed some techniques which might be helpful when other modelling assumptions are considered.

2.3.3 The Solution to the Optimization Problem Using a Linear Program

Coyle and Lin [CL88] have observed that a Boolean stack filter obtained through the solution of the optimization problem discussed in Section 2.3.2 can be determined efficiently by using a linear program (LP). In order to formulate this LP, we define the concept of a randomizing Boolean vector function and introduce its stacking property.

Definition 2.15 *A randomizing Boolean vector function,* $\mathbf{P}_f$*, of a vector Boolean variable* $\mathbf{b}$ *of size* M *is defined as*

$$\mathbf{P}_f = (\ P_f(\mathbf{b}_1),\ P_f(\mathbf{b}_2),\ \ldots,\ P_f(\mathbf{b}_j),\ \ldots,\ P_f(\mathbf{b}_{2^M})\ )\ ,$$

in which $P_f(\mathbf{b}_j) \in [0,1]\ \forall\, j = 1,2,\ldots,2^M$.

Given a randomizing Boolean vector function $\mathbf{P}_f$, a Boolean function (which, in this context, is sometimes called nonrandomizing Boolean function) f can be determined as

$$f(\mathbf{b}_j) = \begin{cases} 1 & \text{if} \quad P_f(\mathbf{b}_j) \geq \mathrm{T}_j \\ 0 & \text{if} \quad P_f(\mathbf{b}_j) < \mathrm{T}_j \end{cases} \qquad \forall\, j = 1,2,\ldots,2^M\ , \tag{2.36}$$

where $\mathrm{T}_j \in (0,1)$. For instance, a simple correspondence from $\mathbf{P}_f$ to f can be defined by taking $\mathrm{T}_j = 0.5\ \forall\, j = 1,2,\ldots,2^M$.

Property 2.3 *A randomizing Boolean vector function* $\mathbf{P}_f$ *is said to satisfy a stacking property if and only if*

$$P_f(\mathbf{b}_\alpha) \geq P_f(\mathbf{b}_\beta) \quad \text{whenever} \quad \mathbf{b}_\alpha \geq \mathbf{b}_\beta\ , \qquad \text{for all} \quad \alpha,\beta \in \{1,2,\ldots,2^M\}\ .$$

Let us now consider a cost function

$$\mathrm{C}' = \sum_{j=1}^{2^M} [\gamma_j \cdot P_f(\mathbf{b}_j) + \beta_j]\ , \tag{2.37}$$

and observe that $\mathrm{C}' \equiv \mathrm{C}(\mathcal{S}_f)$ in the special case when $P_f(\mathbf{b}_j) \in \{0,1\}\ \forall\, j = 1,2,\ldots,2^M$ (see (2.32)). Having introduced the concept of a randomizing Boolean vector function, and having defined the cost function C', we can formulate the following optimization procedure [CL88].

LP Optimization Procedure

Determine a function $\mathbf{P}_f$ *which achieves the following*

$$\textit{minimize} \quad \sum_{j=1}^{2^M} [\gamma_j \cdot P_f(\mathbf{b}_j) + \beta_j] \tag{2.38}$$

subject to the following constraints in the stacking diagram :

$$\textit{for each vertex } \mathbf{b}_j \; : \; 0 \leq P_f(\mathbf{b}_j) \leq 1 \; , \tag{2.39}$$

$$\textit{and for each edge } [(\mathbf{p},0,\mathbf{q}),(\mathbf{p},1,\mathbf{q})] \; : \; P_f(\mathbf{p},1,\mathbf{q}) \geq P_f(\mathbf{p},0,\mathbf{q}) \; . \tag{2.40}$$

It has been shown in [GC91] that all the solutions of the LP optimization problem (Eqns. (2.38-2.40)) must be integer. Besides, as 0 and 1 are the only possible integer solutions, each variable $P_f(\mathbf{b}_j)$ will be either 0 or 1. Therefore, any solution using the LP opimization procedure is a nonrandomizing Boolean filter. Consequently, the LP optimization procedure gives the solution to the original optimization problem given by (2.26) for designing an MMAE stack filter.

The solution to the optimization problem is not necessarily unique. When there are multiple solutions, it means that there are "don't care" situations giving more than one Boolean stack filter.

As shown in [CL88], in the LP design of a stack filter of window size M there are $O(\mathrm{M} \cdot 2^{\mathrm{M}})$ variables and constraints. Even for a relatively small window size, e.g. 4×4 ($\mathrm{M} = 16$), the number of variables and constraints is greater than one million. The computational complexity of the LP approach as well as the assumption that we should have an apriori knowledge regarding the values of the coefficients γ_j's and β_j's render this solution extremely impractical despite its theoretical importance.

2.3.4 The LP-Based Adaptive MMAE Design of Stack Filters

As previously discussed, the LP design approach requires a knowledge of the coefficients γ_j's and β_j's which appear in the expression of the cost function that has to be minimized. As a more practical alternative, an algorithm for the adaptive design of stack filters has been proposed [LSC90]. In this algorithm, an optimal filter is determined by applying a training procedure given a reference (desired) signal and its noise corrupted version. In this approach, even though the computational complexity remains as involved as in the LP design technique, the knowledge of γ_j's and β_j's is not required. This is accomplished by recognizing that when no stacking constraint is imposed on the variables $P_f(\mathbf{b}_j)$'s, then the minimum of the cost function C' given by (2.37) is achieved, if $P_f(\mathbf{b}_j) = 1$ for $\gamma_j < 0$ and $P_f(\mathbf{b}_j) = 0$ for $\gamma_j > 0$ ($\gamma_j = \sum_{\ell=1}^{L-1} [P(0, \mathbf{b}_j|\ell) - P(1, \mathbf{b}_j|\ell)]$). Therefore, in order to control the drift of $\mathbf{P}_f^{(n)}$ (i.e., the randomizing Boolean vector function at the time instant n) in such a way as to move closer to the optimal $\mathbf{P}_{\tilde{f}}$, one can update $\mathbf{P}_f^{(n)}$ based on counting the difference between the frequencies of occurrence of each binary vector $\mathbf{b}_j$ when the desired signal is 1, and when it is 0. Specifically, $P_f(\mathbf{b}_j)$ is increased or decreased when this difference corresponding to $\mathbf{b}_j$ is negative or positive, respectively.

The adaptive algorithm starts with an arbitrary initial guess for the function $\mathbf{P}_f$, $\mathbf{P}_f^{(0)}$. Then, at each time instant n, $\mathbf{P}_f^{(n)}$ is updated as mentioned above. After each updating, the stacking constraint given by (2.40) is enforced by making sure that the vector $\mathbf{P}_f^{(n)}$ remains always inside the polytope described by (2.39) and (2.40). At the end of the training procedure, a Boolean function f is determined from the optimal function $\mathbf{P}_{\tilde{f}}$ by using (2.36) with $T_j = 0.5 \ \forall j = 1, 2, \ldots, 2^M$.

The computational complexity of the adaptive algorithm, however, remains high as in the case of the LP approach, and the resulting optimal filter is useful

when the statistics of the corrupted and desired signals are close enough to the statistics of the reference training sequences.

2.4 Summary

In this chapter, a review of the stack filter theory and the relevant background material has been presented. It has been shown that a stack filter architecture is completely characterized by positive Boolean filters. The stacking property of PBFs allows the development of a theory for optimal design of a stack filter based on the MAE criterion [CL88]. The fundamentals of this theory have been introduced, and it has been shown that in a stack filter architecture, the multilevel mean absolute error can be decomposed into a sum of the mean absolute errors incurred by the Boolean filters at each level. The problem of designing a stack filter which is optimum for the estimation under the MAE criterion was stated in [CL88] as an integer linear program (LP). A new and direct approach for the formulation of this LP has been proposed in this chapter. Finally, the LP-based adaptive technique [LSC90] for the design of a stack filter has been described.

The large number of variables in the LP-based design methodologies, which increases exponentially with the window size, makes them very impractical. In order to reduce the computational expense, new design techniques are needed, with a smaller number of optimization variables.

Chapter 3

Linearly Separable Stack-Like Filters

3.1 Introduction

As shown in Chapter 2, the LP-based design methodologies of stack filters are computationally very expensive, since the number of variables and constraints increases faster than exponentially as a function of the filter's window size. If the design is restricted to the class of stack filters described by linearly separable positive Boolean functions (LSPBFs), the number of variables becomes equal to the window size of the filter. Harja et al [HAN91] have demonstrated that, in the multilevel signal domain, a stack filter described by an LSPBF performs the operation of the weighted order statistic (WOS) filtering. Moreover, the class of WOS filters includes many of the median-related filters of recognized importance in image processing such as rank order filters and the weighted median filters. Therefore, it is reasonable to study the design of WOS filters by using a broader class of stack filter architectures. The goal of this chapter is to extend the class of stack filters described by LSPBFs, and define a stack filter architecture which is described by a linearly separable Boolean

function (LSBF). The properties of this new stack architecture as they relate to WOS filtering, are studied.

3.2 Linearly Separable Boolean Functions

The concept of linearly separable Boolean functions, also referred to as threshold functions [Mur71], is defined as below.

Definition 3.1 *An arbitrary Boolean function* $f(x_0, x_1, \ldots, x_{M-1})$ *is called a linearly separable Boolean function (LSBF) if and only if there exist real numbers* $w_0, w_1, \ldots, w_{M-1}$, *and* w_T *such that*

$$f(x_0, x_1, \ldots, x_{M-1}) = \begin{cases} 1 & \text{if} \quad \sum_{j=0}^{M-1} w_j x_j \geq w_T \\ 0 & \text{otherwise}. \end{cases} \tag{3.1}$$

The numbers $w_0, w_1, \ldots, w_{M-1}$ are called weights, and w_T is called threshold. The expression (3.1) can also be written in a vector form as

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if} \quad \mathbf{w}^T\mathbf{x} \geq w_T \\ 0 & \text{otherwise}, \end{cases} \tag{3.2}$$

where $\mathbf{w} = (w_0, w_1, \ldots, w_{M-1})^T$, and $\mathbf{x} = (x_0, x_1, \ldots, x_{M-1})^T$.

Based on the Definitions 2.4 and 3.1, an LSBF $f(x_0, x_1, \ldots, x_{M-1})$ which is positive in all its variables is called a linearly separable positive Boolean function (LSPBF). The next theorem, which was stated by McNaughton [Mur71], establishes that any linearly separable Boolean function is a unate function.

Theorem 3.1 *A linearly separable Boolean function* $f(\mathbf{x})$ *is a unate function; also, when* f *is dependent on a variable* x_j, *we have* $w_j > 0$ $(w_j < 0)$ *in every structure* $< \mathbf{w}, w_T >$ *for* f, *if* f *is positive (negative) in* x_j.

As a consequence of this theorem, we can formulate the following necessary and sufficient condition for a linearly separable Boolean function to be an LSPBF.

Theorem 3.2 *A linearly separable Boolean function $f(\mathbf{x})$ is a linearly separable PBF if and only if all the weights are positive real numbers (i.e., $w_j \geq 0 \ \forall \ j \ \in \{0, 1, \ldots, M-1\}$).*

Proof:

Necessity (Proof by contradiction): Let us assume that there exists a linearly separable PBF $f(\mathbf{x})$ for which some of the weights are negative (i.e., $w_j < 0$ for some x_j). Based on Theorem 3.1, it follows that f is negative in those variables x_j, which is in contradiction with the definition of a PBF.

Sufficiency: Assuming that all the weights of an LSBF $f(x_0, x_1, \ldots, x_{M-1})$ are positive real numbers, we have to prove that f is an LSPBF. By Definition 3.1, the LSBF f can be expressed as

$$f(x_0, x_1, \ldots, x_{M-1}) = \begin{cases} 1 \quad \text{if} \quad \sum_{j=0}^{M-1} w_j x_j \geq w_T \\ 0 \quad \text{otherwise}. \end{cases} \tag{3.3}$$

We will prove that when all w_j's are positive real numbers, then f satisfies the stacking property, and therefore it is an LSPBF (see also Theorem 2.2). Let us consider two binary vectors of size M, $\mathbf{a}$ and $\mathbf{b}$, with $\mathbf{a} \geq \mathbf{b}$. We can have the following two cases for the vector $\mathbf{a}$.

Case 1: $\sum_{j=0}^{M-1} w_j a_j \geq w_T$, and therefore $f(a_0, a_1, \ldots, a_{M-1}) = 1$. In this case, we see that $f(\mathbf{a}) \geq f(\mathbf{b}) \ \forall \mathbf{b}$.

Case 2: $\sum_{j=0}^{M-1} w_j a_j < w_T$, and therefore $f(a_0, a_1, \ldots, a_{M-1}) = 0$. As $b_j \leq a_j \ \forall j \in \{0, 1, \ldots, M-1\}$ and w_j's are all positive, it follows that $\sum_{j=0}^{M-1} w_j b_j \leq \sum_{j=0}^{M-1} w_j a_j < w_T$. Consequently, in this case, $f(\mathbf{b}) = f(\mathbf{a}) = 0$.

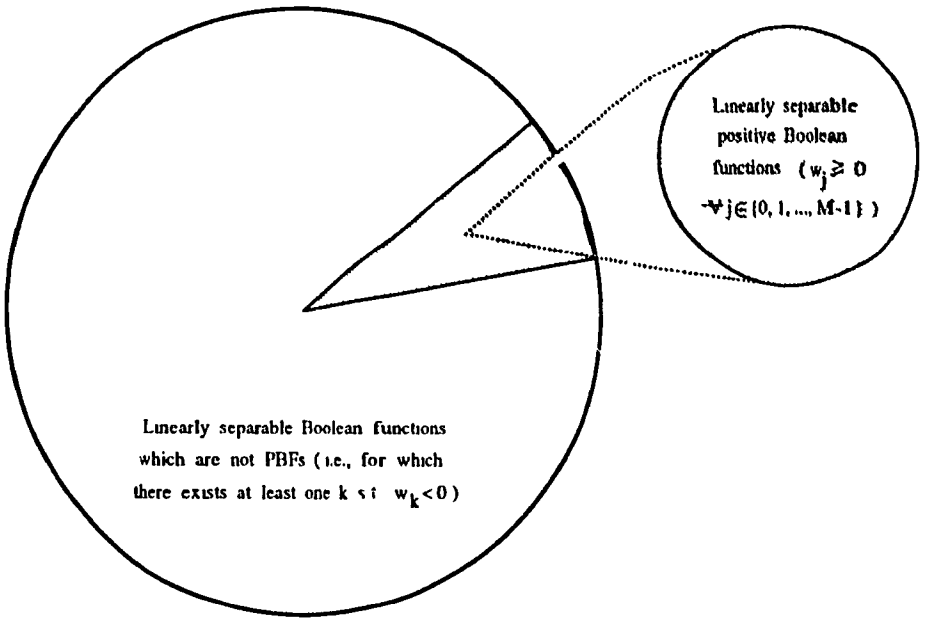


Figure 3.1: A necessary and sufficient condition for a linearly separable Boolean function to be a PBF is that all w_j's should be positive.

Now, we can conclude that function $f(\mathbf{x})$ satisfies the stacking property, i.e.,

$$f(\mathbf{a}) \geq f(\mathbf{b}) \text{ whenever } \mathbf{a} \geq \mathbf{b},$$

and therefore it is a PBF.

□

Figure 3.1 illustrates the result of Theorem 3.2. As a consequence of Definition 3.1 and Theorem 3.2, an LSBF can be expressed as given by the following theorem.

Theorem 3.3 *Any linearly separable Boolean function $f(\mathbf{x})$ given by (3.1) can be expressed in the form:*

$$f(x_0, x_1, \ldots, x_{M-1}) = \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} w_j^{\dagger} \cdot (\overline{g_j \oplus x_j}) \geq w_T^{\dagger} \\ 0 & \text{otherwise}, \end{cases} \tag{3.4}$$

where all the weights $w_j^\dagger$ *are positive real numbers, that is* $w_j^\dagger = |w_j|$, *the gate coefficients* g_j*'s are related to the signs of* w_j*'s by* $g_j = \mathrm{sgn}[w_j]$, *which means that* g_j *is equal to 1 when* f *is positive in the variable* x_j, *and it is equal to 0 when* f *is negative in* x_j, *and the threshold is given by* $w_T^\dagger = w_T - \sum_{j=0}^{M-1} \overline{g}_j w_j$.

Proof:

Without restricting the generality, the proof can be reduced to the case when f is negative in only one variable, say x_k. If f is positive in some variable x_j ($w_j > 0$), then, in (3.1) we take $w_j^\dagger = w_j$ and replace x_j by $\overline{g_j \oplus x_j}$ with $g_j = 1$. We have:

$$\overline{g_j \oplus x_j} = \overline{g_j \overline{x}_j + \overline{g}_j x_j} = x_j .$$

From Theorem 3.2, when an LSBF f is negative in the variable x_k, $w_k < 0$. Then

$$\begin{aligned} f(x_0, x_1, \ldots, x_k, \ldots, x_{M-1}) &= \begin{cases} 1 & \text{if } \left(\sum_{j=0, j\neq k}^{M-1} w_j^\dagger x_j \right) + w_k x_k \geq w_T \\ 0 & \text{otherwise} \end{cases} \\ &= \begin{cases} 1 & \text{if } \left(\sum_{j=0, j\neq k}^{M-1} w_j^\dagger x_j \right) + (-w_k)\overline{x}_k + w_k(x_k + \overline{x}_k) \geq w_T \\ 0 & \text{otherwise} \end{cases} \\ &= \begin{cases} 1 & \text{if } \left(\sum_{j=0, j\neq k}^{M-1} w_j^\dagger x_j \right) + (-w_k)\overline{x}_k \geq w_T + (-w_k) \\ 0 & \text{otherwise}, \end{cases} \end{aligned}$$

and therefore we have:

$$f(x_0, x_1, \ldots, x_k, \ldots, x_{M-1}) = \begin{cases} 1 & \text{if } \left[\sum_{j=0, j\neq k}^{M-1} w_j^\dagger \cdot (\overline{g_j \oplus x_j}) \right] + w_k^\dagger \cdot (\overline{g_k \oplus x_k}) \geq w_T^\dagger \\ 0 & \text{otherwise}, \end{cases} \tag{3.5}$$

where $g_j = 1 \ \forall j \in \{0, 1, \ldots, k-1, k+1, \ldots, M-1\}$, $g_k = 0$, $w_k^\dagger = -w_k > 0$, and $w_T^\dagger = w_T - w_k$. Now, (3.5) can be expressed in the same form as (3.4).

□

As an example, the function

$$f(x_0, x_1, x_2, x_3, x_4) = \begin{cases} 1 & \text{if } \; x_0 - 2x_1 + 2x_2 - x_3 - x_4 \geq -1 \\ 0 & \text{otherwise}, \end{cases} \tag{3.6}$$

can be expressed as

$$\begin{aligned} f(x_0, x_1, x_2, x_3, x_4) &= \begin{cases} 1 & \text{if } \; x_0 + 2\overline{x}_1 + 2x_2 + \overline{x}_3 + \overline{x}_4 \geq 3 \\ 0 & \text{otherwise} \end{cases} \\ &= \begin{cases} 1 & \text{if } \; \overline{1 \oplus x_0} + 2 \cdot (\overline{0 \oplus x_1}) + 2 \cdot (\overline{1 \oplus x_2}) + \overline{0 \oplus x_3} + \overline{0 \oplus x_4} \geq 3 \\ 0 & \text{otherwise}. \end{cases} \end{aligned}$$

In view of Theorem 3.3, an LSBF can be equivalently defined as follows.

Definition 3.2 *An arbitrary Boolean function $f(x)$ is said to be a linearly separable Boolean function if it can be expressed in the form*

$$f(x_0, x_1, \ldots, x_{M-1}) = \begin{cases} 1 & \text{if } \; \sum_{j=0}^{M-1} w_j^{\dagger} \cdot (\overline{g_j \oplus x_j}) \geq w_T^{\dagger} \\ 0 & \text{otherwise}, \end{cases} \tag{3.7}$$

where all the weights $w_j^{\dagger}$'s are positive real numbers, and the gate coefficients, g_j's have values of 0 or 1.

Muroga has shown in [Mur71] that any LSBF can be converted into an LSPBF. Using Theorem 3.3, we can restate the result of Muroga as given below.

Theorem 3.4 *Any linearly separable Boolean function $f(x_0, x_1, \ldots, x_{M-1})$ which is not a PBF (i.e., for which there exists at least one g_j which is equal to 0) can be converted into an LSPBF $f^{\dagger}(y_0, y_1, \ldots, y_{M-1})$ by the transformation:*

$$f^{\dagger}(y_0, y_1, \ldots, y_{M-1}) = f(x_0, x_1, \ldots, x_{M-1}) \Big|_{x_j = g_j \oplus \overline{y}_j \quad \forall j \in \{0, 1, \ldots, M-1\}} \tag{3.8}$$

Proof:

By substituting for $x_j = g_j \oplus \overline{y}_j$ in the expression (3.7) of $f(x_0, x_1, \ldots, x_{M-1})$, we obtain

$$f^{\dagger}(y_0, y_1, \ldots, y_{M-1}) = \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} w_j \cdot \left(\overline{(g_j \oplus g_j) \oplus \overline{y}_j} \right) \geq w_T^{\dagger} \\ 0 & \text{otherwise} \end{cases}$$

$$= \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} w_j \cdot \left(\overline{0 \oplus x_j} \right) \geq w_T^{\dagger} \\ 0 & \text{otherwise} \end{cases}$$

$$= \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} w_j y_j \geq w_T^{\dagger} \\ 0 & \text{otherwise} . \end{cases}$$

□

As an example, the function

$$f^{\dagger}(y_0, y_1, y_2, y_3, y_4) = \begin{cases} 1 & \text{if } y_0 + 2y_1 + 2y_2 + y_3 + y_4 \geq 3 \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

is the linearly separable PBF obtained from $f(x_0, x_1, x_2, x_3, x_4)$ given in (3.6) by applying the transformation described by Theorem 3.4.

3.3 Stack Filters Characterized by LSPBFs

It has been shown that in multilevel signal domain, a stack filter described by an LSPBF performs the operation of the weighted order statistic filtering [HAN91]. In the following, the definition of WOS filters is given, and it is shown that rank order, weighted median and standard median filters can be realized using special cases of linearly separable stack filter architectures [HAN91].

In order to give a mathematical description for WOS filtering, an operation called replication and denoted by "$\Diamond$" has been introduced in [HAN91]. The replication of a real number R_j by the positive integer w_j is defined as a vector

$$\mathbf{R}_j = R_j \Diamond w_j = \overbrace{(R_j, R_j, \ldots, R_j)}^{w_j \text{ times}} . \tag{3.10}$$

Now, a formal definition of a WOS filter can be given as follows [HAN91].

Definition 3.3 *The output of a WOS filter at the discrete time n is obtained by the following procedure:*

(a) Replicate each input sample, $\mathrm{R}(n-j)$ $(j = 0, 1, \ldots, \mathrm{M}-1)$, appearing in the filter's input window (which is assumed to be of size M) at time n, by a given positive integer w_j called weight.

(b) Sort the resulting vector of $\sum_{j=0}^{\mathrm{M}-1} \mathrm{w}_j$ elements.

(c) Choose the w_T-th largest value (where w_T denotes a positive integer called threshold) from the sorted vector.

In view of the above definition, the output of a WOS filter is given by

$$\begin{aligned} \mathrm{WOS}(n) \;=\; & \mathrm{w}_\mathrm{T}\text{-th largest element in} \\ & \{\, \mathrm{R}(n) \Diamond \mathrm{w}_0,\ \mathrm{R}(n-1) \Diamond \mathrm{w}_1,\ \ldots,\ \mathrm{R}(n-\mathrm{M}+1) \Diamond \mathrm{w}_{\mathrm{M}-1} \,\}\ . \end{aligned} \tag{3.11}$$

In the following, it is shown that rank order, weighted median, and standard median filters can be realized using special cases of linearly separable stack filter architectures [HAN91]. For each case, the format of the corresponding LSPBF is specified.

A. Rank Order Filters

Definition 3.4 *In the multilevel signal domain, the output of a rank order filter is obtained by sorting the input samples $\mathbf{R}_\mathrm{M}(n)$ and then choosing the w_Tth largest sample.*

Thus, the class of rank order filters is a special case of WOS filters for which all the weights w_j's are set equal to 1, and only the threshold w_T can be varied. A stack

filter defined by a positive Boolean function $f(\mathbf{x})$ performs the operation of rank order filtering, if and only if there exists a threshold w_T such that

$$f(x_0, x_1, \ldots, x_{M-1}) = \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} x_j \geq w_T \\ 0 & \text{otherwise.} \end{cases} \tag{3.12}$$

B. Weighted Median Filters

Definition 3.5 *In the multilevel signal domain, the output of an weighted median filter is obtained by replicating each sample* $R(n-j)$ *appearing in the input window* $\mathbf{R}_M(n)$ *by the number of the corresponding weight* w_j*, then sorting the resulting array of* $\sum_{j=0}^{M-1} w_j$ *elements* [1]*, and finaly choosing the median value of the sorted array.*

Thus, weighted median filters are a special case of WOS filters, where $w_T = (1 + \sum_{j=0}^{M-1} w_j)/2$ and $\sum_{j=0}^{M-1} w_j$ is odd. A stack filter defined by a PBF $f(\mathbf{x})$ performs the operation of weighted median filtering if and only if there exista weights w_j's such that can be expressed as

$$f(x_0, x_1, \ldots, x_{M-1}) = \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} w_j x_j \geq (1 + \sum_{j=0}^{M-1} w_j)/2 \\ 0 & \text{otherwise.} \end{cases} \tag{3.13}$$

C. Standard Median Filters

Definition 3.6 *In the multilevel signal domain, the output of a standard median filter is obtained by sorting the samples in the input window,* $\mathbf{R}_M(n)$*, and choosing the centermost value, the number of input samples being odd.*

The standard median filter is a special case of a WOS filter, where all the weights w_j's are equal to 1 and $w_T = \frac{M+1}{2}$. A stack filter defined by a PBF $f(\mathbf{x})$ performs

[1] It is assumed that $\sum_{j=0}^{M-1} w_j$ is an odd number.

the operation of standard median filtering if and only if

$$f(x_0, x_1, \ldots, x_{M-1}) = \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} x_j \geq (1+M)/2 \\ 0 & \text{otherwise.} \end{cases} \tag{3.14}$$

3.4 Linearly Separable Stack-Like Filter Architectures

In this section, the definition of a linearly separable stack-like (LSSL) filter is formulated based on introducing a new type of threshold decomposition of an L-level signal appearing in a window of size M. It is shown that in an LSSL filter architecture, although the sequences in the binary input windows do not satisfy the stacking property in a strict sense, the binary output signals do. Due to this property, the fundamentals of the optimality theory that has been developed in [CL88] (see Section 2.3) still remain valid for the class of LSSL filters. Finally, it is shown that in the multilevel signal domain, an LSSL filter architecture performs the operation of WOS filtering.

3.4.1 Generalized Threshold Decomposition of a Multilevel Finite Sequence

The generalized threshold decomposition of an L-level sequence of size M is defined as follows.

Definition 3.7 *Let* $\mathbf{R}_M(n) = (R(n), R(n-1), \ldots, R(n-M+1))$ *denote an* L*-level discrete-time sequence appearing in the window of size* M *at instant* n. *Given a binary vector* $\mathbf{g} = (g_0, g_1, \ldots, g_{M-1})^T$, *the corresponding generalized threshold decomposition of* $\mathbf{R}_M(n)$ *is the set of* $(L-1)$ *binary sequences called* $\mathbf{g}$*-threshold signals,* $\mathbf{gr}_1(n), \mathbf{gr}_2(n), \ldots, \mathbf{gr}_{L-1}(n)$, *with*

$$\mathbf{gr}_\ell(n) = (\, gr_\ell(n), gr_\ell(n-1), \ldots, gr_\ell(n-M+1) \,)^T, \tag{3.15}$$

whose element $\mathrm{gr}_\ell(n-j)$ *is defined by*

$$\mathrm{gr}_\ell(n-j) = \overline{\mathrm{g}_j \oplus \mathrm{r}_\ell(n-j)}\,, \tag{3.16}$$

where

$$\mathrm{r}_\ell(n-j) = \begin{cases} 1 & \textit{if } \mathrm{R}(n-j) \geq \ell \\ 0 & \textit{if } \mathrm{R}(n-j) < \ell\,, \end{cases} \tag{3.17}$$

for $\ell = 1, 2, \ldots, \mathrm{L}-1$, *and* $j = 0, 1, \ldots, \mathrm{M}-1$.

An example of the generalized threshold decomposition of a multilevel window sequence with $\mathrm{L} = 8$ and $\mathrm{M} = 5$, given the template $\mathbf{g} = (1,0,1,0,0)$, is shown in Figure 3.2.

As the elements of the g-threshold vector $\mathbf{gr}_\ell(n)$ are given by (3.16), it is convenient to use the following simplified notation

$$\mathbf{gr}_\ell(n) = \overline{\mathbf{g} \oplus \mathbf{r}_\ell(n)}\,, \tag{3.18}$$

where $\oplus$ denotes a vectorial exclusive OR operation, in which every element of the resulting vector is given by the scalar exclusive OR operation performed between the corresponding elements of the vectors $\mathbf{g}$ and $\mathrm{r}_\ell(n)$. The binary vector $\overline{\mathbf{g} \oplus \mathbf{r}_\ell(n)}$, is understood as the vector obtained by taking the complement of each element of $\mathbf{g} \oplus \mathrm{r}_\ell(n)$. In (3.18), $\mathbf{r}_\ell(n)$ denotes a vector whose elements are given by the binary samples at the level ℓ in the threshold decomposition of an L-level window sequence of size M, that is,

$$\mathbf{r}_\ell(n) = \left(\, \mathrm{r}_\ell(n),\ \mathrm{r}_\ell(n-1),\ \ldots,\ \mathrm{r}_\ell(n-\mathrm{M}+1) \,\right)^{\mathrm{T}}.$$

3.4.2 The Definition of a Linearly Separable Stack-Like Filter Architecture

It was seen in Chapter 2, that a stack filter is characterized by a PBF. In the case when the PBF is linearly separable, the corresponding architecture can be called

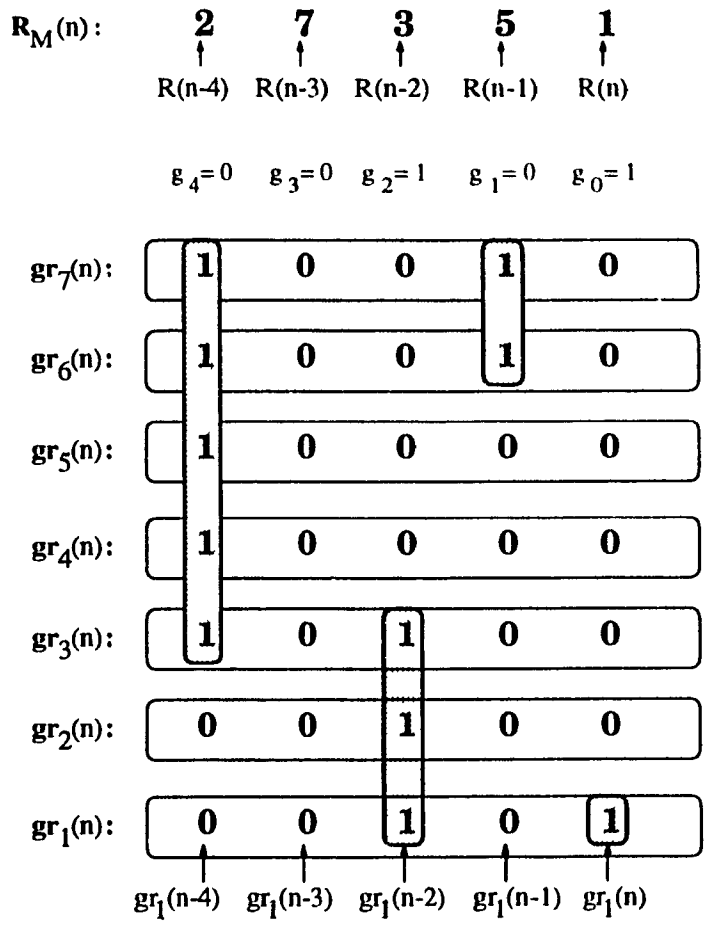


Figure 3.2: Generalized threshold decomposition of a multilevel sequence with L = 8 and M = 5, given the template g = (1, 0, 1, 0, 0).

linearly separable stack (LSS) architecture. Now, we will develop an architecture characterized by an LSBF that will be called linearly separable stack-like (LSSL) architecture. Starting from an LSBF

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if} \quad \sum_{j=0}^{M-1} w_j x_j \geq w_T \\ 0 & \text{otherwise}\,, \end{cases} \tag{3.19}$$

in this architecture, a multilevel input sequence $\mathbf{R}_M(n)$ appearing in the filter's window of size M at the discrete time n is first decomposed into a set of binary sequences $\mathbf{gr}_\ell(n)$ $\ell = 1, 2, \ldots, L-1$, by using the template

$$\mathbf{g} = (\, \mathrm{sgn}\,[\,w_0\,],\ \mathrm{sgn}\,[\,w_1\,],\ \ldots,\ \mathrm{sgn}\,[\,w_{M-1}\,]\,)^T\,, \tag{3.20}$$

then a filtering is performed via the LSBF given by (3.19) on each of the L − 1 g-threshold signals, that is,

$$f(\mathbf{gr}_\ell) = \begin{cases} 1 & \text{if} \quad \sum_{j=0}^{M-1} w_j \cdot \mathrm{gr}_\ell(n-j) \geq w_T \\ 0 & \text{otherwise} \end{cases} \qquad \ell = 1, 2, \ldots, L-1\,, \tag{3.21}$$

and finally, all the binary results are added together yielding the multilevel output. Denoting by $\mathcal{SL}_f(\mathbf{R}_M(n))$ the output of a stack-like filter operating on an input process $\mathbf{R}_M(n)$, we have

$$\mathcal{SL}_f(\mathbf{R}_M(n)) = \sum_{\ell=1}^{L-1} f(\mathbf{gr}_\ell(n))\,. \tag{3.22}$$

As an example, Figure 3.3 illustrates a stack-like filter architecture described by the linearly separable Boolean function given in (3.6).

3.4.3 Generalized Stacking Property of Boolean Functions

As stated in Section 2.2.4 (see Property 2.2), a Boolean filter $f(\mathbf{x})$ of size M is said to satisfy the stacking property if and only if

$$f(\mathbf{b}_\alpha) \geq f(\mathbf{b}_\beta) \quad \text{whenever} \quad \mathbf{b}_\alpha \geq \mathbf{b}_\beta\,, \qquad \text{for all} \quad \alpha, \beta \in \{1, 2, \ldots, 2^M\}\,,$$

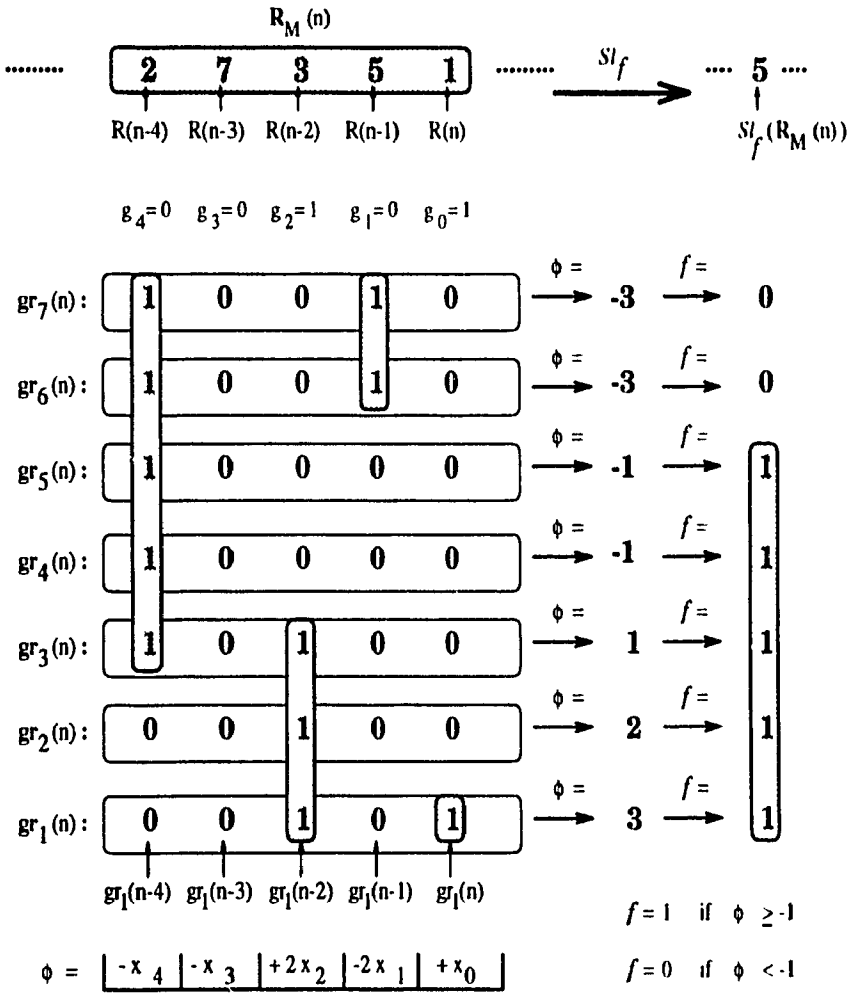


Figure 3.3: An example of a stack-like filter architecture described by a linearly separable Boolean function $f(x_0, x_1, x_2, x_3, x_4)$, which takes a value of 1 if and only if $x_0 - 2x_1 + 2x_2 - x_3 - x_4 \geq -1$.

where $\mathbf{b}_\alpha$ and $\mathbf{b}_\beta$ denote Boolean input vectors of size M. It has also been mentioned in the same section that the necessary and sufficient condition for a Boolean filter to satisfy the stacking property is that the filter function be a PBF [Gil54], [Mur71].In Section 3.2, it has been shown that the necessary and sufficient condition for an LSBF to be a linearly separable PBF is that all the weights be positive. Therefore, if an LSBF is not a PBF, then it does not satisfy the stacking property. As a result, if the PBF in a certain stack filter architecture is replaced by an LSBF which is not a PBF, the binary output signals will no longer satisfy the stacking property. This is illustrated by the example of Figure 3.4, where the LSBF given by (3.6) has been used on each level of the architecture. The input sequence satisfies the stacking property, but the output sequence doesn't, since the filter function is not a PBF. However, it can be seen from Figure 3.3, that in a stack-like filter architecture, although the binary input signals do not satisfy the stacking property in a strict sense, the binary output signals do.

In this section, the generalized stacking property of a Boolean function with respect to a given template vector $\mathbf{g}$ is stated, and it is shown that any LSBF satisfies this property with respect to the template

$$\mathbf{g} = (\operatorname{sgn}[w_0], \operatorname{sgn}[w_1], \ldots, \operatorname{sgn}[w_{M-1}])^T .$$

Property 3.1 *Given a template vector* $\mathbf{g} = (g_0, g_1, \ldots, g_{M-1})^T$, *a Boolean filter (function)* $f(x_0, x_1, \ldots, x_{M-1})$ *of size* M, *is said to satisfy the generalized stacking property with respect to* $\mathbf{g}$ *if and only if*

$$f(\overline{\mathbf{g} \oplus \mathbf{b}_\alpha}) \geq f(\overline{\mathbf{g} \oplus \mathbf{b}_\beta}) \quad \text{whenever} \quad \mathbf{b}_\alpha \geq \mathbf{b}_\beta , \qquad \text{for all} \quad \alpha, \beta \in \{1, 2, \ldots, 2^M\} .$$

Theorem 3.5 *Any linearly separable Boolean function*

$$f(x_0, x_1, \ldots, x_{M-1}) = \begin{cases} 1 & \text{if} \quad \sum_{j=0}^{M-1} w_j x_j \geq w_T \\ 0 & \text{otherwise} \end{cases} \tag{3.23}$$

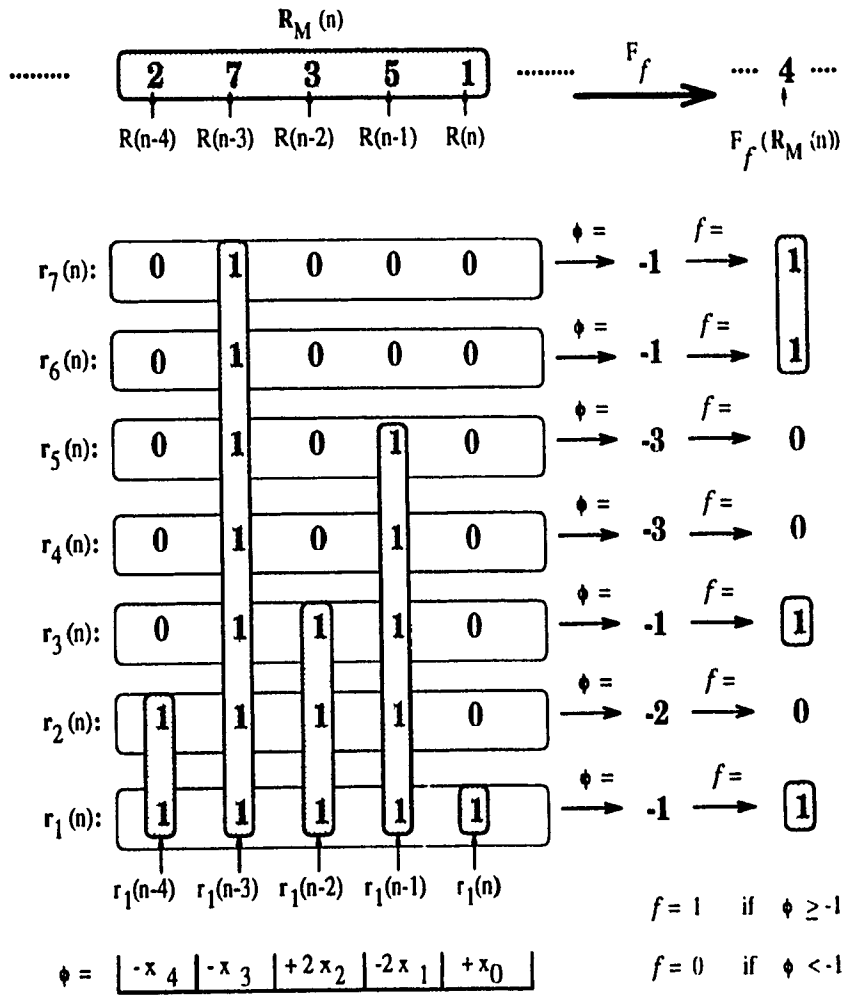


Figure 3.4: An example showing that if in a stack filter architecture the characteristic PBF is replaced by an LSBF which is not positive, then the binary output signals no longer satisfy the stacking property.

satisfies the generalized stacking property with respect to the template

$$\mathbf{g} = (\operatorname{sgn}[w_0], \operatorname{sgn}[w_1], \ldots, \operatorname{sgn}[w_{M-1}])^T . \tag{3.24}$$

Proof:

By Theorem 3.3, any LSBF $f(\mathbf{x})$ can be expressed in the form (3.4), i.e.,

$$f(x_0, x_1, \ldots, x_{M-1}) = \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} w_j^\dagger \cdot (\overline{g_j \oplus x_j}) \geq w_T^\dagger \\ 0 & \text{otherwise}, \end{cases} \tag{3.25}$$

where the weights $w_j^\dagger$ are given by $w_j^\dagger = |w_j|$, $g_j = \operatorname{sgn}[w_j]$, and the threshold is given by $w_T^\dagger = w_T - \sum_{j=0}^{M-1} \overline{g}_j w_j$. Substituting for $\mathbf{x} = \overline{\mathbf{g} \oplus \mathbf{y}}$ in (3.25), we get

$$\begin{aligned} f(\overline{\mathbf{g} \oplus \mathbf{y}}) &= \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} w_j^\dagger \cdot \left(\overline{g_j \oplus \overline{g_j \oplus y_j}}\right) \geq w_T^\dagger \\ 0 & \text{otherwise} \end{cases} \\ &= \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} w_j^\dagger y_j \geq w_T^\dagger \\ 0 & \text{otherwise}. \end{cases} \end{aligned}$$

Therefore, since all w_j's are positive, we have that $f(\overline{\mathbf{g} \oplus \mathbf{b}_\alpha}) \geq f(\overline{\mathbf{g} \oplus \mathbf{b}_\beta})$ whenever $\mathbf{b}_\alpha \geq \mathbf{b}_\beta$

□

As shown in Section 2.2.5, due to the stacking property of PBFs in a stack filter architecture, the binary outputs at each time instant n will have a structure in which a stack of 0's is piled on top of a stack of 1's. As a consequence of Theorem 3.5, a stack-like filter architecture defined by an LSBF also preserves the stacking property at the filter's output. Therefore, the fundamentals of the optimality theory based on the MMAE criterion, that has been developed for the class of stack filters [CL88] (see Section 2.3), remain valid for the class of LSSL filters as well.

3.4.4 WOS Filtering in a Linearly Separable Stack-Like Architecture

In this section, it is shown that each stack-like filter architecture which is characterized by an LSBF is equivalent to some stack filter architecture described by an LSPBF.

Theorem 3.6 *A stack-like filter architecture which is described by the LSBF*

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} w_j x_j \geq w_T \\ 0 & \text{otherwise}, \end{cases} \tag{3.26}$$

is equivalent to the stack filter architecture characterized by the LSPBF

$$f^\dagger(\mathbf{x}) = \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} w_j^\dagger x_j \geq w_T^\dagger \\ 0 & \text{otherwise}, \end{cases} \tag{3.27}$$

where,

$$w_j^\dagger = |\, w_j \,|, \qquad j = 0, 1, \ldots, M-1, \tag{3.28}$$

and

$$w_T^\dagger = w_T - \sum_{j=0}^{M-1} \overline{g}_j w_j, \tag{3.29}$$

with

$$g_j = \operatorname{sgn}[\, w_j \,], \qquad j = 0, 1, \ldots, M-1. \tag{3.30}$$

Proof:

The output of a linearly separable stack-like filter is given by (3.22) as

$$S\ell_f(\mathbf{R}_M(n)) = \sum_{\ell=1}^{L-1} f(\overline{\mathbf{g} \oplus \mathbf{r}_\ell(n)}). \tag{3.31}$$

By Theorem 3.3, the LSBF (3.26) can be expressed in the form

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} w_j^\dagger \cdot (\overline{g_j \oplus x_j}) \geq w_T^\dagger \\ 0 & \text{otherwise}, \end{cases} \tag{3.32}$$

where $w_j^\dagger = |w_j|$, $w_T^\dagger = w_T - \sum_{j=0}^{M-1} \overline{g}_j w_j$, and $g_j = \mathrm{sgn}[w_j]$. Substituting for $\mathbf{x} = \overline{\mathbf{g} \oplus \mathbf{r}_\ell(n)}$ in (3.32), we have

$$\begin{aligned} f(\overline{\mathbf{g} \oplus \mathbf{r}_\ell(n)}) &= \begin{cases} 1 & \text{if } \displaystyle\sum_{j=0}^{M-1} w_j^\dagger \cdot \left(\overline{\overline{g_j \oplus \overline{g_j \oplus r_\ell(n-j)}}} \right) \geq w_T^\dagger \\ 0 & \text{otherwise} \end{cases} \\ &= \begin{cases} 1 & \text{if } \displaystyle\sum_{j=0}^{M-1} w_j^\dagger \cdot r_\ell(n-j) \geq w_T^\dagger \\ 0 & \text{otherwise}. \end{cases} \end{aligned} \tag{3.33}$$

The output of the stack filter described by $f^\dagger(\mathbf{x})$ in (3.27) is given by (2.16) as

$$\mathcal{S}_{f^\dagger}(\mathbf{R}_M(n)) = \sum_{\ell=1}^{L-1} f^\dagger(\mathbf{r}_\ell(n)), \tag{3.34}$$

with

$$f^\dagger(\mathbf{r}_\ell(n)) = \begin{cases} 1 & \text{if } \displaystyle\sum_{j=0}^{M-1} w_j^\dagger \cdot r_\ell(n-j) \geq w_T^\dagger \\ 0 & \text{otherwise}. \end{cases} \tag{3.35}$$

From (3.31), (3.33), (3.34), and (3.35), we conclude that

$$\mathcal{SL}_f(\mathbf{R}_M(n)) = \mathcal{S}_{f^\dagger}(\mathbf{R}_M(n)). \tag{3.36}$$

□

For example, the stack-like filter architecture shown in Figure 3.3, which is described by the function f given in (3.6), i.e.,

$$f(x_0, x_1, x_2, x_3, x_4) = \begin{cases} 1 & \text{if } x_0 - 2x_1 + 2x_2 - x_3 - x_4 \geq -1 \\ 0 & \text{otherwise}, \end{cases} \tag{3.37}$$

is equivalent to the stack filter architecture described by

$$f^\dagger(x_0, x_1, x_2, x_3, x_4) = \begin{cases} 1 & \text{if } x_0 + 2x_1 + 2x_2 + x_3 + x_4 \geq 3 \\ 0 & \text{otherwise}, \end{cases} \tag{3.38}$$

and illustrated in Figure 3.5.

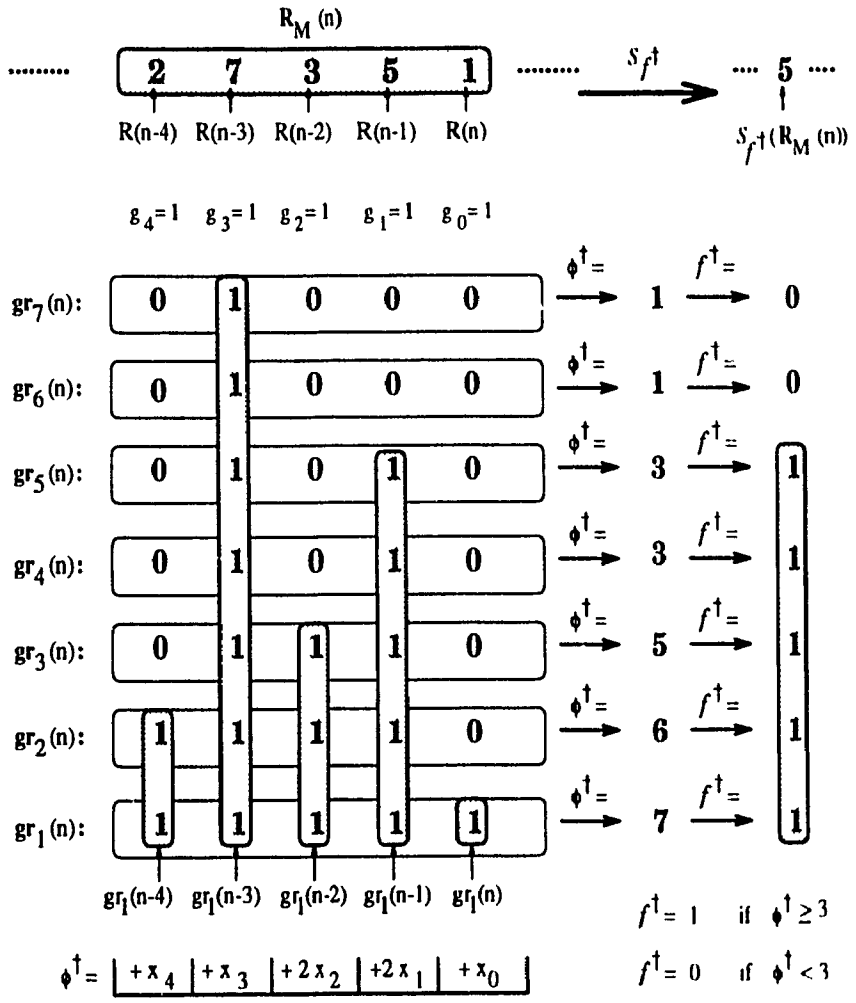


Figure 3.5: A stack filter which performs the same filtering operation as the stack-like filter shown in Figure 3.3. Note that a stack filter described by an LSPBF is also a particular case of linearly separable stack-like filter for which the template vector g has all the elements equal to 1.

Since stack filters described by LSPBFs have the physical interpretation of WOS filtering, based on Theorem 3.6, we conclude that the stack-like filter architectures described by LSBFs have the same physical interpretation. Although linearly separable stack-like filters do not perform a filtering operation other than that of WOS filtering, it will be shown in the next chapter that this new type of architecture is very useful in the design of WOS filters.

3.5 Summary

In this chapter, in the framework of generalized threshold decomposition of multi-level sequences, a new type of filter configuration called linearly separable stack-like architecture has been developed. An LSSL filter architecture is characterized by an LSBF, and has been shown to preserve the stacking property at the binary-level output. This feature stems from the generalized stacking property of linearly separable Boolean functions. The LSSL filters have been shown to perform the operation of WOS filtering. In Chapter 4, an adaptive algorithm for the design of WOS filters using the LSSL architecture of this chapter will be developed. It is expected that this design would yield better results, since the weights in an LSSL architecture are not restricted to assume only positive or zero values.

Chapter 4

The LMAE Adaptive Design of WOS Filters Based on the LSSL Architecture and Application to Image Processing

4.1 Introduction

The theory of linear FIR adaptive filtering is well developed [WS85], [Hay91], and its powerful results constitute an attractive theoretical framework for the design of median-related filters. For instance, Bovik et al [BHM83] have combined the properties of averaging and median filters in a class of nonlinear filters called order statistic filters or L-filters, whose output is a linear combination of the order statistics of the input sequence. The theory of order statistic filters and their relationship to linear FIR filters were investigated by Longbotham and Bovik [LB89]. A design algorithm in which the goal is to find an order statistic filter which minimizes the mean square error criterion has been developed in [PB88]. The order statistic filters combine the rank order operations with linear operations. It has been shown in

[Gab89] that the design of such filters has a major drawback in that the optimization is in fact reduced to optimizing the linear part of the filter. The definition of stack filters was an effort to correct this situation [Gab89].

The class of stack filters is better suited for estimation under MAE criterion. A heuristic configuration of stack filters in which the PBF in the threshold decomposition architecture is replaced by a neural network has been proposed for the design of WOS filters [YAN92], [AHL92]. With the help of this configuration, the design is performed by using a constrained binary-level LMS algorithm for adjusting the weights of the neural network. These weights are allowed to assume only positive or zero values, so that the neuron represents an LSPBF. It has been shown that this heuristic approach gives satisfactory experimental results. Recently, a theoretical formulation for the adaptive design of LSS filters has been developed [YAN93], by using a procedure which is similar to Widrow's derivation of the LMS algorithm [WS85], [Hay91]. This design involves a least mean absolute error (LMAE) algorithm. It is to be noted that this approach is referred to as the LMAE design instead of MMAE design, since the cost function that is minimized is not strictly the mean absolute error as given by (2.23). In the daptive LMAE design of LSS filters, the weights which become negative during the training procedure are reset to zero. This is a constrained optimization, in the sense that the weights can assume either positive or zero values, but they are not allowed to assume negative values. However, there could be situations in which the mean absolute error could be further reduced if the weights are allowed to assume any value, positive or negative.

In this chapter, a theory for the adaptive LMAE design of linearly separable stack-like filters is developed. The LMAE criterion is introduced following an approach which is similar to that given in [YAN93]. It is shown that the LMAE of an LSSL filter is close to the minimum mean absolute error. An adaptive LMAE algorithm for the design of LSSL filters is derived, and the implementation of this new design by using a binary-level LMS algorithm is discussed. Experimental results

in image restoration are provided in order to demonstrate the performance of the algorithm. The significance of the proposed approach is that within the framework of stack-like filters, it is possible to develop a less constrained adaptive LMAE design algorithm than that given in [YAN93], and which still enjoys the implementation simplicity of stack filters. The less constrained design is achieved by allowing the weights to assume real values instead of just positive or zero values. This flexibility in the weight values is used to explore the possibility of lowering the mean absolute error compared to that given by LSS filters.

4.2 The LSSL Filter Architecture and MMAE Estimation in WOS Filtering

The fundamentals of the optimality theory which has been developed for the design of a stack filter based on the MAE criterion [CL88] still remain valid for the case of LSSL filters. Following a procedure similar to the one developed by Coyle in [CL88] and summarized in Section 2.3.1, the estimation problem for the class of linearly separable stack-like filters can be stated using a scheme shown in Figure 4.1. The process $\mathrm{R}(n)$ which is received at the input of an LSSL filter is assumed to be a corrupted version of some desired process $\mathrm{S}(n)$ through an operation $g(\cdot,\cdot)$. The corruption may be caused either by a noise process $\mathrm{N}(n)$ or by some intentional operation, such as a modulation scheme. At each time instant n, the output of the LSSL filter is an estimate, denoted by $\hat{\mathrm{S}}(n)$, of the desired process $\mathrm{S}(n)$. This estimate is based on the sequence $\mathbf{R}_{\mathrm{M}}(n)$ observed in the input window of the LSSL filter architecture and it is given by

$$\hat{\mathrm{S}}(n) = \mathcal{SL}_f(\mathbf{R}_{\mathrm{M}}(n))\,. \tag{4.1}$$

The mean absolute error between the desired signal, $\mathrm{S}(n)$, and the estimated signal, $\hat{\mathrm{S}}(n)$, at the time instant n, is defined to be the cost of using the LSSL filter

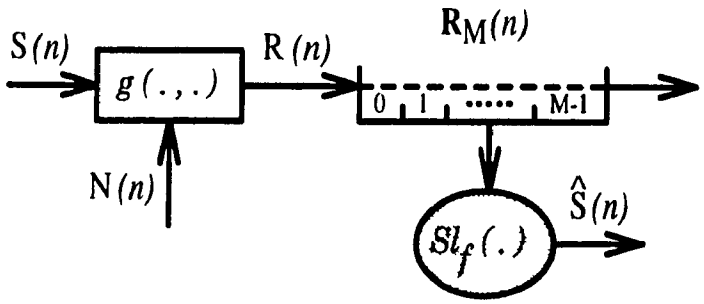


Figure 4.1: The structure of the estimation problem for the class of stack-like filters.

$\mathcal{S}\ell_f$ at time n, and it can be expressed as

$$C_n(\mathcal{S}\ell_f) = E\left[\,\left|\, S(n) - \mathcal{S}\ell_f(\mathbf{R}_M(n)) \,\right|\,\right] . \tag{4.2}$$

If we make the assumption that the signal and noise processes are jointly stationary, then the cost function $C_n(\mathcal{S}\ell_f)$ has the same value at all time instants n. Therefore, in the subsequent discussion, the subscript n will not be used for the cost function. The optimal LSSL filter for estimation under the mean absolute error criterion is the filter for which $C(\mathcal{S}\ell_f)$ is minimum.

By using the technique given in [CL88] and described in Section 2.3.1, the cost function $C(\mathcal{S}\ell_f)$ can be decomposed into the sum of the mean absolute errors incurred at each binary level ℓ, that is,

$$\begin{aligned} C(\mathcal{S}\ell_f) &= \sum_{\ell=1}^{L-1} E[\,|\, e_\ell(n) \,|\,] \\ &= \sum_{\ell=1}^{L-1} E[\,|\, s_\ell(n) - f(\mathbf{gr}_\ell(n)) \,|\,] \ . \end{aligned} \tag{4.3}$$

Now, the procedure proposed in Section 2.3.2 can be used in order to express the cost function $C(\mathcal{S}\ell_f)$ into a form similar to the one given by (2.32), which is

$$C(\mathcal{S}\ell_f) = \sum_{j=1}^{2^M} [\gamma_j \cdot f(\mathbf{b}_j) + \beta_j] , \tag{4.4}$$

with $\gamma_j = \sum_{\ell=1}^{L-1}[P(0,\mathbf{b}_j|\ell) - P(1,\mathbf{b}_j|\ell)] = \sum_{\ell=1}^{L-1}[P(\mathbf{b}_j|\ell) - 2P(1,\mathbf{b}_j|\ell)]$, and $\beta_j = \sum_{\ell=1}^{L-1} P(1,\mathbf{b}_j|\ell)$. Note that $P(0,\mathbf{b}_j|\ell) + P(1,\mathbf{b}_j|\ell) = P(\mathbf{b}_j|\ell)$, and based on the

definition of an LSSL filter architecture, we have

$$P(1,\mathbf{b}_j|\ell) = P(1,\overline{\mathbf{g}\oplus\mathbf{b}_j}|\ell) \quad \text{and} \quad P(0,\mathbf{b}_j|\ell) = P(0,\overline{\mathbf{g}\oplus\mathbf{b}_j}|\ell)\,, \tag{4.5}$$

for all 2^M possible g-vectors. Since $P(1,\mathbf{b}_j|\ell) = P(1|\mathbf{b}_j,\ell)P(\mathbf{b}_j|\ell)$, we can also express γ_j as

$$\begin{aligned}\gamma_j &= P(\mathbf{b}_j|\ell)\left[(L-1) - 2\textstyle\sum_{\ell=1}^{L-1} P(1|\mathbf{b}_j,\ell)\right] \\ &= 2(L-1)\left[\tfrac{1}{2} - \tfrac{1}{L-1}\textstyle\sum_{\ell=1}^{L-1} P(1|\mathbf{b}_j,\ell)\right]. \end{aligned}\tag{4.6}$$

The optimization problem to determine an MMAE WOS filter using an LSSL architecture can be stated as to determine a Boolean filter $\tilde{f}$ of window size M, which achieves

$$\begin{cases} C(\mathcal{S}\ell_{\tilde{f}}) = \min\limits_{f\,\in\,\mathcal{F}_M} \sum\limits_{j=1}^{2^M}[\gamma_j\cdot f(\mathbf{b}_j)+\beta_j]\,, \\ \text{subject to the constraint that } \tilde{f} \text{ should be linearly separable.} \end{cases}\tag{4.7}$$

In order to obtain the solution to this optimization problem, the constraint of linear separability has to be imposed. In view of the difficulties involved in imposing this constraint, alternative solutions for the design of WOS filters using LSSL architecture must be found. The LMAE design using LSSL filters introduced in the next section is one suboptimal solution to this problem. It will be shown that this design would yield a filter that is close to the solution of the unconstrained problem of minimizing the cost function $C(\mathcal{S}\ell_f)$ given by (4.4) It can be shown easily that the solution to the unconstrained optimization problem is given by

$$\begin{aligned} f(\mathbf{b}_j) &= \begin{cases} 1 & \text{if } \gamma_j \le 0 \\ 0 & \text{otherwise} \end{cases} \\ &= \begin{cases} 1 & \text{if } \frac{1}{(L-1)}\sum_{\ell=1}^{L-1} P(1|\mathbf{b}_j,\ell) \ge \frac{1}{2} \\ 0 & \text{otherwise}\,, \end{cases} \end{aligned}\tag{4.8}$$

for all b_j's. Another advantage of the LMAE approach is that it is well suited for adaptive design of WOS filters. This point will be further discussed in the next section.

4.3 The LMAE Criterion and the Design of WOS Filters Using the LSSL Architecture

In the LMAE design of WOS filters using linearly separable stack-like architectures, the cost function that is minimized is given by

$$\begin{aligned} \tilde{C}(\mathbf{w}) &= \sum_{\ell=1}^{L-1} E\left[\,|\,\tilde{e}_\ell(n)\,|^2\,\right] \\ &= \sum_{\ell=1}^{L-1} E\left[\,\left|\,s_\ell(n) - \mathbf{w}^T \mathbf{gr}_\ell(n)\,\right|^2\right], \end{aligned} \tag{4.9}$$

where

$$\mathbf{w} = (\,w_0,\ w_1,\ \ldots,\ w_{M-1}\,)^T, \tag{4.10}$$

and $\mathbf{gr}_\ell(n)$ denotes the vector

$$\begin{aligned} \mathbf{gr}_\ell(n) &= \left(\overline{g_0 \oplus r_\ell(n)},\ \overline{g_1 \oplus r_\ell(n-1)},\ \ldots,\ \overline{g_{M-1} \oplus r_\ell(n-M+1)}\right)^T \\ &= \overline{\mathbf{g} \oplus r_\ell(n)}\,. \end{aligned}$$

Since the absolute value of the binary level error in (4.3) can assume only values of 0 and 1, the quantity under the expected value operator in (4.3) can be squared as in (4.9). Therefore, there is a similarity between the cost functions $C(\mathcal{SL}_f)$ given by (4.3) and $\tilde{C}(\mathbf{w})$ given by (4.9). However, in view of the fact that the term $f(\mathbf{gr}_\ell(n))$ in (4.3) has a binary value, while $\mathbf{w}^T\mathbf{gr}_\ell(n)$ in (4.9) can assume real values, the two cost functions are essentially different.

It has been shown in [YAN93] that an LMAE stack filter is always sufficiently close to an MMAE stack filter. Following an argument similar to the one given in [YAN93], we will now demonstrate that if the minimum of $\tilde{C}(\mathbf{w})$ is achieved for a set

of weights $\mathbf{w}_o = (\mathrm{w}_{o,0}, \mathrm{w}_{o,1}, \ldots, \mathrm{w}_{o,M-1})$, then the LSSL architecture characterized by the LSBF

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if} \quad \mathbf{w}_o^{\mathrm{T}}\mathbf{x} \geq \dfrac{1}{2} \\ 0 & \text{if} \quad \text{otherwise}\,, \end{cases} \tag{4.11}$$

is close to the function f given by (4.8), which achieves the unconstrained minimum of the cost function $\mathrm{C}(\mathcal{S}\ell_f)$ given by (4.3). The WOS filter using an LSSL architecture characterized by the Boolean function (4.11) is designated as an LMAE WOS filter.

The binary random variable $\mathrm{s}_\ell(n)$ specifies the desired output corresponding to the input $\mathbf{r}_\ell(n)$. For a given $\mathbf{r}_\ell(n) = \mathbf{b}_\alpha$, there is a probability $P(1|\mathbf{b}_\alpha, \ell)$ that $\mathrm{s}_\ell(n) = 1$ and a probability $P(0|\mathbf{b}_\alpha, \ell) = 1 - P(1|\mathbf{b}_\alpha, \ell)$ that $\mathrm{s}_\ell(n) = 0$. The expected value of the random variable $\mathrm{s}_\ell(n)$ is given by

$$\begin{aligned} E_{\mathrm{s}_\ell}[\mathrm{s}_\ell(n)] &= 1 \cdot P(1|\mathbf{b}_\alpha, \ell) + 0 \cdot P(0|\mathbf{b}_\alpha, \ell) \\ &= P(1|\mathbf{b}_\alpha, \ell)\,, \end{aligned} \tag{4.12}$$

where $E_{\mathrm{s}_\ell}[\cdot]$ denotes the expectation over the distribution of s_ℓ for the given $\mathbf{b}_\alpha$. Hence, as outlined in [YAN93], $\mathrm{s}_\ell(n)$ may be regarded as being given by the sum of a deterministic component, $P(1|\mathbf{b}_\alpha, \ell)$, and a zero-mean random component $\eta_\ell(n)$, i.e.,

$$\mathrm{s}_\ell(n) = P(1|\mathbf{b}_\alpha, \ell) + \eta_\ell(n)\,. \tag{4.13}$$

Indeed, we have that $E_{\eta_\ell}[\eta_\ell(n)] = E_{\mathrm{s}_\ell}[\mathrm{s}_\ell(n)] - P(1|\mathbf{b}_\alpha, \ell) = 0$. We also note that

$$\begin{aligned} E_{\mathrm{s}_\ell}\left[(\mathrm{s}_\ell(n))^2\right] &= 1^2 \cdot P(1|\mathbf{b}_\alpha, \ell) + 0^2 \cdot P(0|\mathbf{b}_\alpha, \ell) \\ &= P(1|\mathbf{b}_\alpha, \ell)\,. \end{aligned} \tag{4.14}$$

By expressing $\eta_\ell(n)$ as in (4.13), and then taking the expected value of its squre, we get

$$E_{\eta_\ell}\left[(\eta_\ell(n))^2\right] = E_{\mathrm{s}_\ell}\left[(\mathrm{s}_\ell(n))^2\right] - 2P(1|\mathbf{b}_\alpha, \ell)E_{\mathrm{s}_\ell}[\mathrm{s}_\ell(n)] + P^2(1|\mathbf{b}_\alpha, \ell) \tag{4.15}$$

Substituting from (4.12) and (4.14) in (4.15) yields

$$E_{\eta_\ell}\left[(\eta_\ell(n))^2\right] = P(1|\mathbf{b}_\alpha,\ell) - P^2(1|\mathbf{b}_\alpha,\ell)\,. \tag{4.16}$$

The cost function $\tilde{C}(\mathbf{w})$ can be expressed as

$$\begin{aligned}\tilde{C}(\mathbf{w}) &= \sum_{\ell=1}^{L-1} E\left\{\left[s_\ell(n) - \mathbf{w}^T\mathbf{g}\mathbf{r}_\ell(n)\right]^2\right\} \\ &= \sum_{\ell=1}^{L-1} E_{\mathbf{b}_\alpha} E_{s_\ell}\left\{\left[s_\ell(n) - \mathbf{w}^T\mathbf{g}\mathbf{b}_\alpha\right]^2\right\}\,,\end{aligned} \tag{4.17}$$

where $\mathbf{g}\mathbf{b}_\alpha = \overline{\mathbf{g} \oplus \mathbf{b}_\alpha}$. Substituting for $s_\ell(n)$ as given by (4.13) in (4.17), we get

$$\begin{aligned}\tilde{C}(\mathbf{w}) &= \sum_{\ell=1}^{L-1} E_{\mathbf{b}_\alpha} E_{\eta_\ell}\left\{\left[\left(P(1|\mathbf{b}_\alpha,\ell) - \mathbf{w}^T\mathbf{g}\mathbf{b}_\alpha(n)\right) + \eta_\ell(n)\right]^2\right\} \\ &= \sum_{\ell=1}^{L-1} E_{\mathbf{b}_\alpha}\left\{\left[P(1|\mathbf{b}_\alpha,\ell) - \mathbf{w}^T\mathbf{g}\mathbf{b}_\alpha\right]^2\right\} \\ &\quad + 2\sum_{\ell=1}^{L-1} E_{\mathbf{b}_\alpha}\left\{\left[P(1|\mathbf{b}_\alpha,\ell) - \mathbf{w}^T\mathbf{g}\mathbf{b}_\alpha\right] E_{\eta_\ell}\left[\eta_\ell(n)\right]\right\} \\ &\quad + \sum_{\ell=1}^{L-1} E_{\eta_\ell}\left[(\eta_\ell(n))^2\right]\,.\end{aligned} \tag{4.18}$$

Taking into account that $E_{\eta_\ell}[\eta_\ell(n)] = 0$ and substituting for $E_{\eta_\ell}\left[(\eta_\ell(n))^2\right]$ as given by (4.16) in (4.18), $\tilde{C}(\mathbf{w})$ becomes

$$\begin{aligned}\tilde{C}(\mathbf{w}) &= \sum_{\ell=1}^{L-1} E_{\mathbf{b}_\alpha}\left\{\left[P(1|\mathbf{b}_\alpha,\ell) - \mathbf{w}^T\mathbf{g}\mathbf{b}_\alpha)\right]^2\right\} \\ &\quad + \sum_{\ell=1}^{L-1}\left[P(1|\mathbf{b}_\alpha,\ell) - P^2(1|\mathbf{b}_\alpha,\ell)\right]\,.\end{aligned} \tag{4.19}$$

In order to obtain an LMAE WOS filter, the weight vector $\mathbf{w}$ must be evaluated for which the cost function $\tilde{C}(\mathbf{w})$ given by (4.19) is minimized. Note that in the expression for $\tilde{C}(\mathbf{w})$ in (4.19), the second term does not depend on $\mathbf{w}$, and therefore, the minimum of $\tilde{C}(\mathbf{w})$ is achieved when the first term in (4.19) is minimum. Since the expected value in the first term of (4.19) is given by

$$E_{\mathbf{b}_\alpha}\left\{\left[P(1|\mathbf{b}_\alpha,\ell) - \mathbf{w}^T\mathbf{g}\mathbf{b}_\alpha)\right]^2\right\} = \sum_{j=1}^{2^M}\left\{\left[P(1|\mathbf{b}_j,\ell) - \mathbf{w}^T\mathbf{g}\mathbf{b}_j)\right]^2 \cdot P(\mathbf{b}_j|\ell)\right\}\,, \tag{4.20}$$

it can be stated that the absolute minimum of $\tilde{C}(\mathbf{w})$ is achieved if there exists a set of weights $\mathbf{w}_o = (w_{o,0}, w_{o,1}, \ldots, w_{o,M-1})$ so that

$$\mathbf{w}_o^T \mathbf{g}\mathbf{b}_j - P(1|\mathbf{b}_j, \ell) \approx 0 \ , \quad \text{for all } j = 1, 2, \ldots, 2^M \text{ and } \ell = 1, 2, \ldots, L-1 \quad (4.21)$$

Adding the relations (4.21) for all $\ell = 1, 2, \ldots, L-1$, and then dividing by $L-1$, we obtain

$$\mathbf{w}_o^T \mathbf{g}\mathbf{b}_j \approx \frac{1}{L-1} \sum_{\ell=1}^{L-1} P(1|\mathbf{b}_j, \ell) \ , \quad \text{for all } j = 1, 2, \ldots, 2^M . \quad (4.22)$$

Consequently, an LSSL filter which is characterized by (4.11) with the weights $\mathbf{w}_o$ given by (4.22), achieves the absolute minimum of $\tilde{C}(\mathbf{w})$. In view of (4.22), this LSSL filter is also close to the function f given by (4.8) which achieves the unconstrained minimum of $C(\mathcal{S}\ell_f)$ given by (4.3). The LMAE design yields a linearly separable Boolean filter. When this filter is used for signal estimation, the mean absolute error is close to the minimum absolute error when the signal is estimated by a Boolean filter designed without imposing the constraint of separability.

4.4 Adaptive LMAE Design of WOS Filters Using the LSSL Architecture

In this section, an adaptive LMAE algorithm for the design of WOS filters using the LSSL architecture, in which negative weights are allowed, is developed. This is a less constrained design method as compared with the design of WOS filters using the LSS architecture, that has been proposed in [YAN93]. A binary-level LMS implementation of the new design algorithm is also discussed.

4.4.1 Derivation of the Proposed Algorithm

In order to derive an adaptive LMAE algorithm for the design of WOS filters using the LSSL architecture, a procedure which is similar to Widrow's derivation of the

LMS algorithm [WS85], [Hay91], is followed. The cost $\tilde{C}(\mathbf{w})$ in (4.9) is a quadratic function in w_j's, as given by

$$\begin{aligned}\tilde{C}(\mathbf{w}) &= \sum_{\ell=1}^{L-1} E\left[\,|\,\tilde{e}_\ell(n)\,|^2\,\right] \\ &= \sum_{\ell=1}^{L-1} E\left[\,\Big(s_\ell(n) - \mathbf{w}^T\mathbf{gr}_\ell(n)\Big)^2\,\right]\,, \end{aligned} \tag{4.23}$$

where

$$\mathbf{w} = (\,w_0,\, w_1,\, \ldots,\, w_{M-1}\,)^T\,, \tag{4.24}$$

and

$$\mathbf{gr}_\ell(n) = \Big(\overline{g_0 \oplus r_\ell(n)},\, \overline{g_1 \oplus r_\ell(n-1)},\, \ldots,\, \overline{g_{M-1} \oplus r_\ell(n-M+1)}\Big)^T\,. \tag{4.25}$$

Therefore, the dependence of the cost function $\tilde{C}(\mathbf{w})$ on the elements of the vector $\mathbf{w}$ can be visualized as a bowl-shaped surface with a unique minimum. An adaptive algorithm attempts to find this minimum by applying succesive corrections to $\mathbf{w}$ in a direction opposite to that of the gradient vector of $\tilde{C}(\mathbf{w})$. Denoting this gradient by $\nabla\tilde{C}(\mathbf{w})$, we have

$$\nabla\tilde{C}(\mathbf{w}) = -2\sum_{\ell=1}^{L-1} E\left[\,\Big(s_\ell(n) - \mathbf{w}^T\mathbf{gr}_\ell(n)\Big)\,\mathbf{gr}_\ell(n)\,\right]\,. \tag{4.26}$$

The weight vector is updated by using the following recursive relation

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \frac{1}{2}\mu\left[\,\nabla\tilde{C}(\mathbf{w})\,\right]\,. \tag{4.27}$$

In (4.27), μ is a positive real-valued constant called step-size, and the factor $\frac{1}{2}$ is used merely for convenience. Substituting for $\nabla C(\mathbf{w})$ as given by (4.26) in (4.27), we get

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu\sum_{\ell=1}^{L-1} E\left[\,\Big(s_\ell(n) - \mathbf{w}^T\mathbf{gr}_\ell(n)\Big)\,\mathbf{gr}_\ell(n)\,\right]\,. \tag{4.28}$$

Since we do not have the knowledge regarding the value of the mathematical expectation on the right side of (4.28), we use the instantaneous value at time instant n of

the quantity under the E operator as an estimate of the expected value. Therefore, we get the following expression for the adaptive LMAE algorithm:

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \mu \sum_{\ell=1}^{L-1} \left[\left(s_\ell(n) - \hat{\mathbf{w}}^{\mathrm{T}}(n) \hat{\mathbf{gr}}_\ell(n) \right) \hat{\mathbf{gr}}_\ell(n) \right] . \tag{4.29}$$

In (4.29), $\hat{\mathbf{gr}}_\ell(n)$ denotes the estimate of $\mathbf{gr}_\ell(n)$ at the iteration n, that is,

$$\hat{\mathbf{gr}}_\ell(n) = \left(\overline{\hat{g}_0(n) \oplus r_\ell(n)}, \overline{\hat{g}_1(n) \oplus r_\ell(n-1)}, \ldots, \overline{\hat{g}_{M-1}(n) \oplus r_\ell(n-M+1)} \right)^{\mathrm{T}} , \tag{4.30}$$

with

$$\hat{g}_j(n) = \mathrm{sgn}\left[\hat{w}_j(n) \right] , \qquad j = 0, 1, \ldots, M-1 , \tag{4.31}$$

Based on the discussion in Section 4.3, the estimate of an optimal linearly separable Boolean f'ter at time $n+1$ is given by

$$\hat{f}(\mathbf{x}, n+1) = \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} \hat{w}_j(n+1) \cdot x_j \geq \frac{1}{2} \\ 0 & \text{otherwise} , \end{cases} \tag{4.32}$$

and in view of the result of Theorem 3.6, the equivalent optimal linearly separable Boolean filter is given by

$$\hat{f}^{\dagger}(\mathbf{x}, n+1) = \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} \hat{w}_j^{\dagger}(n+1) \cdot x_j \geq \hat{w}_{\mathrm{T}}^{\dagger}(n+1) \\ 0 & \text{otherwise} , \end{cases} \tag{4.33}$$

where

$$\hat{w}_j^{\dagger}(n+1) = \left| \hat{w}_j(n+1) \right| , \qquad j = 0, 1, \ldots, M-1 , \tag{4.34}$$

and

$$\hat{w}_{\mathrm{T}}^{\dagger}(n+1) = \frac{1}{2} - \sum_{j=0}^{M-1} \overline{\hat{g}_j(n+1)} \cdot \hat{w}_j(n+1) , \tag{4.35}$$

with

$$\hat{g}_j(n+1) = \mathrm{sgn}\left[\hat{w}_j(n+1) \right] , \qquad j = 0, 1, \ldots, M-1 . \tag{4.36}$$

Figure 4.2 depicts a functional illustration of the LMAE design of WOS filters using the LSSL architecture. In this figure, each radius of the larger circle, represents

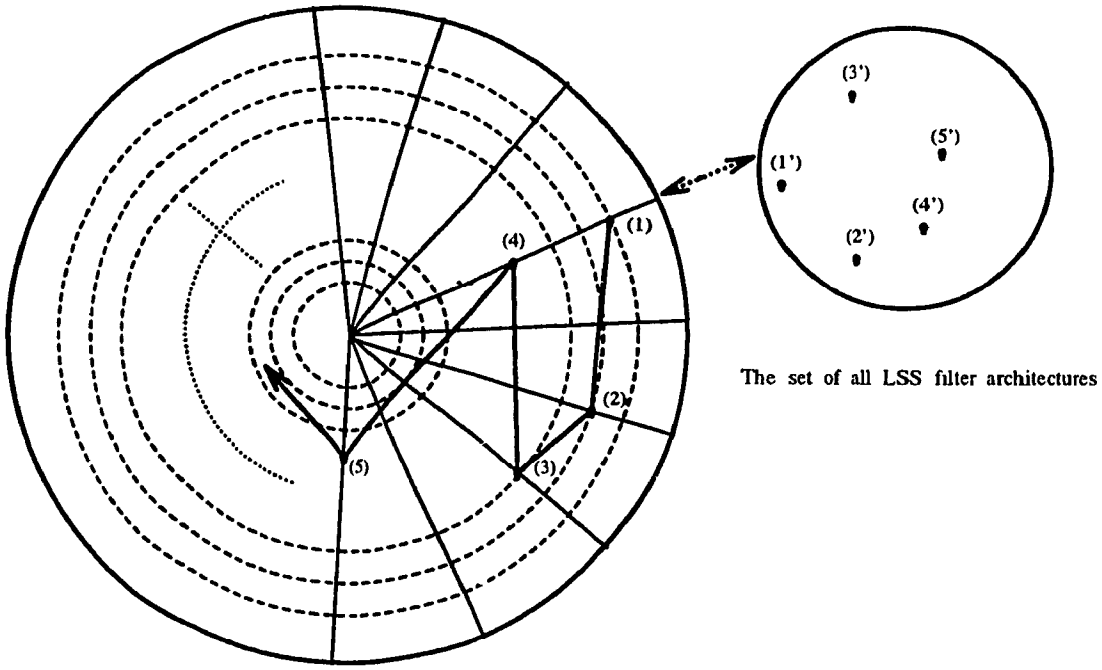


Figure 4.2: Illustrations of the correspondence between the LSSL and LSS architectures, and of the steps of the adaptive design of a WOS filter using the LSSL architecture.

a subclass of LSSL architectures which are characterized by the same $\mathbf{g}$ template vector. Each dashed circle represents a contour connecting all the LSSL architectures which are equivalent to a certain LSS filter architecture, and thus they all have the same MAE. These contours are assumed to be arranged such that the MAE decreases while moving towards the centre of the larger circle. The set of all WOS filtering operations can be obtained simply by taking all the LSSL architectures corresponding to any one of the radii. The domain of the LSS filter architectures is shown by the smaller circle. All the points on a given dashed contour in the bigger circle map to a single point in the smaller circle, in the sense that all the LSSL filter architectures corresponding to the points involved give the same MAE and perform the same WOS filtering operation. The bold line (1),(2),(3),(4),(5),..., represents the trajectory of a design in the domain of LSSL architectures. The points (1'),(2'),(3'),(4'),(5'),..., are the images of points (1),(2),(3),(4),(5),..., during this design process.

4.4.2 An Implementation of the Proposed Adaptive Design by Using a Binary-Level LMS Algorithm

In this section, it is shown that the adaptive LMAE algorithm (4.29) can be implemented by using the LMS algorithm at each binary level in the generalized threshold decomposition architecture, and by imposing the constraint that at each time instant the weights should assume the same values at all the levels. This observation could be very useful for a hardware implementation of the adaptive LMAE design of WOS filters using the LSSL architecture.

In order to minimize the cost function $\tilde{C}(\mathbf{w})$ given by (4.23), we use the LMS algorithm of Widrow and Stearns [WS85], [Hay91] at each binary level ℓ, that is,

$$\begin{aligned} \hat{\mathbf{w}}_\ell(n+1) &= \hat{\mathbf{w}}_\ell(n) + \mu_0\, \tilde{e}_\ell(n)\, \hat{\mathbf{gr}}_\ell(n) \\ &= \hat{\mathbf{w}}_\ell(n) + \mu_0 \left(s_\ell(n) - \hat{\mathbf{w}}_\ell^T(n)\hat{\mathbf{gr}}_\ell(n)\right) \hat{\mathbf{gr}}_\ell(n)\,, \end{aligned} \tag{4.37}$$

for $\ell = 1, 2, \ldots, \mathrm{L}-1$. But, since at iteration n in a stack-like filter architecture the same LSBF is used at each level ℓ, we must impose the condition that

$$\hat{\mathbf{w}}_\ell(n) = \hat{\mathbf{w}}(n) \qquad \forall\, \ell = 1, 2, \ldots, \mathrm{L}-1\,. \tag{4.38}$$

Therefore, (4.37) becomes

$$\hat{\mathbf{w}}_\ell(n+1) = \hat{\mathbf{w}}(n) + \mu_0 \left(\mathrm{s}_\ell(n) - \hat{\mathbf{w}}^{\mathrm{T}}(n)\hat{\mathbf{gr}}_\ell(n)\right) \hat{\mathbf{gr}}_\ell(n)\,, \quad \ell = 1, 2, \ldots, \mathrm{L}-1\,. \tag{4.39}$$

Adding up the equations (4.39) for ℓ from 1 to $\mathrm{L}-1$, and taking into account that the weight-vector $\hat{\mathbf{w}}$ is the same for all ℓ, we get

$$\sum_{\ell=1}^{\mathrm{L}-1} \hat{\mathbf{w}}_\ell(n+1) = (\mathrm{L}-1)\,\hat{\mathbf{w}}(n) + \mu_0 \sum_{\ell=1}^{\mathrm{L}-1} \left[\left(\mathrm{s}_\ell(n) - \hat{\mathbf{w}}^{\mathrm{T}}(n)\hat{\mathbf{gr}}_\ell(n)\right) \hat{\mathbf{gr}}_\ell(n)\right]\,. \tag{4.40}$$

Dividing both sides of (4.40) by $\mathrm{L}-1$ and using the notation

$$\mu = \frac{\mu_0}{\mathrm{L}-1}\,, \tag{4.41}$$

we obtain the following expression for the adaptive algorithm:

$$\frac{1}{\mathrm{L}-1} \sum_{\ell=1}^{\mathrm{L}-1} \hat{\mathbf{w}}_\ell(n+1) = \hat{\mathbf{w}}(n) + \mu \sum_{\ell=1}^{\mathrm{L}-1} \left[\left(\mathrm{s}_\ell(n) - \hat{\mathbf{w}}^{\mathrm{T}}(n)\hat{\mathbf{gr}}_\ell(n)\right) \hat{\mathbf{gr}}_\ell(n)\right]\,. \tag{4.42}$$

Consequently, by taking

$$\hat{\mathbf{w}}(n+1) = \frac{1}{\mathrm{L}-1} \sum_{\ell=1}^{\mathrm{L}-1} \hat{\mathbf{w}}_\ell(n+1)\,, \tag{4.43}$$

it is possible to implement the algorithm (4.29) by using the serial procedure (4.39). Note that the derivation of the LMAE algorithm given in this subsection is not only justifying a serial implementation for the algorithm, but also suggesting an approach for proving the convergence of the algorithm (4.29). Specifically, it can be stated that the algorithm (4.29) converges if the LMS algorithm converges at each binary level, under the constraint that, at each time instant, the weights have to assume the same values at all the levels. Therefore, a procedure similar to that given by Widrow and Stearns [WS85], [Hay91] can be used for proving the convergence of (4.29).

4.5 Experimental Results

So far, we have restricted our discussion of WOS filtering and stack architectures using 1-D sequences. This has been done in order to keep all the derivations simple and to make the exposition clear and concise. However, as it is the case with almost all median-related filters, an important application of the LSSL filter architecture is for the problem of image reconstruction from the observation data corrupted by impulsive noise. Since the experiments reported in this section are concerned with this specific type of application, we will first briefly introduce the notations that are used in extending the 1-D concepts to the 2-D case.

4.5.1 The Extension of 1-D LSSL Filtering to the 2-D Case

In this subsection, the extension of 1-D LSSL filtering to the 2-D case is presented, by considering 3×3 filters. We will also be considering the 2-D computational procedure for which the input and output masks are as shown in Figure 4.3. Thus, the binary samples in the input window at the time instant n are given by

$$\begin{bmatrix} \mathrm{x}(n+1,n-1) & \mathrm{x}(n,n-1) & \mathrm{x}(n-1,n-1) \\ \mathrm{x}(n+1,n) & \mathrm{x}(n,n) & \mathrm{x}(n-1,n) \\ \mathrm{x}(n+1,n+1) & \mathrm{x}(n,n+1) & \mathrm{x}(n-1,n+1) \end{bmatrix},$$

and are denoted as

$$\begin{bmatrix} \mathrm{x}_0 & \mathrm{x}_1 & \mathrm{x}_2 \\ \mathrm{x}_3 & \mathrm{x}_4 & \mathrm{x}_5 \\ \mathrm{x}_6 & \mathrm{x}_7 & \mathrm{x}_8 \end{bmatrix}.$$

The output at each time instant n is calculated by using an LSBF

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \sum_{j=0}^{M-1} \mathrm{w}_j \mathrm{x}_j \geq \frac{1}{2} \\ 0 & \text{otherwise}. \end{cases} \tag{4.44}$$

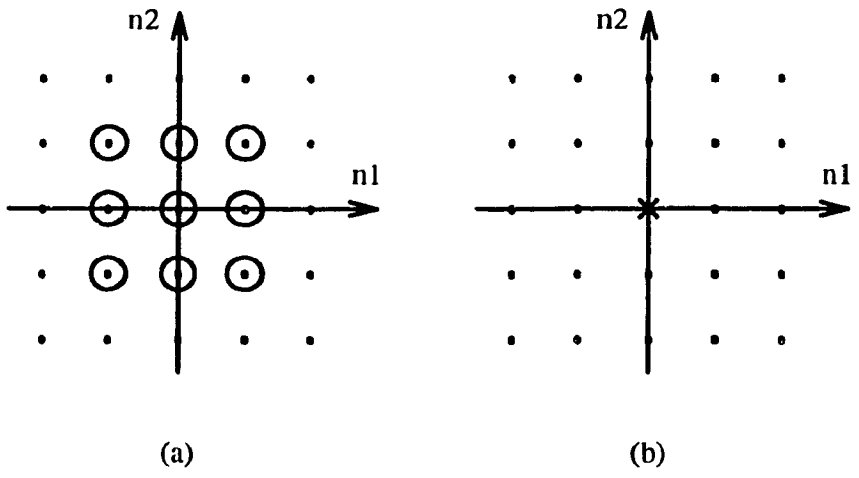


Figure 4.3: (a) Input and (b) output masks corresponding to a computational procedure which realizes a 3×3 2-D filter. This type of computational procedure has been used in all the experiments reported in Section 4.5.2.

When using the design method developed in Section 4.4.2, we determine the optimal w_j's in the LMAE sense. In order to conform the notation of the filter's weight with that of the input samples in the window, the weights are denoted as

$$\begin{bmatrix} w_0 & w_1 & w_2 \\ w_3 & w_4 & w_5 \\ w_6 & w_7 & w_8 \end{bmatrix}.$$

The LSS filter architecture which is equivalent to an LSSL architecture characterized by (4.44), is described by an LSPBF for which the weights are given by $w_j^\dagger = |w_j|$ $(j = 1, 2, \ldots, 8)$, and $w_T^\dagger = \frac{1}{2} - \sum_{j=0}^{8} \overline{g_j} w_j$ $(g_j = sgn[w_j])$.

4.5.2 An Application of the Proposed WOS Filter Design in Image Reconstruction

In this section, some experimental results are presented to illustrate the performance of the new adaptive LMAE design method developed in Section 4.4.2. An application of this design for restoring images that are corrupted by impulsive noise, is

considered. The WOS filters designed using the LSSL architecture are compared with those obtained by using the LSS architecture. In the latter case, the common practice of resetting to zero, the weights which become negative during training, has been followed [YAN92], [AHL92], [YAN93]. It has been observed that by using the LSSL architecture, it is possible to achieve lower, or at least the same errors, as compared to those obtained with the LSS architecture.

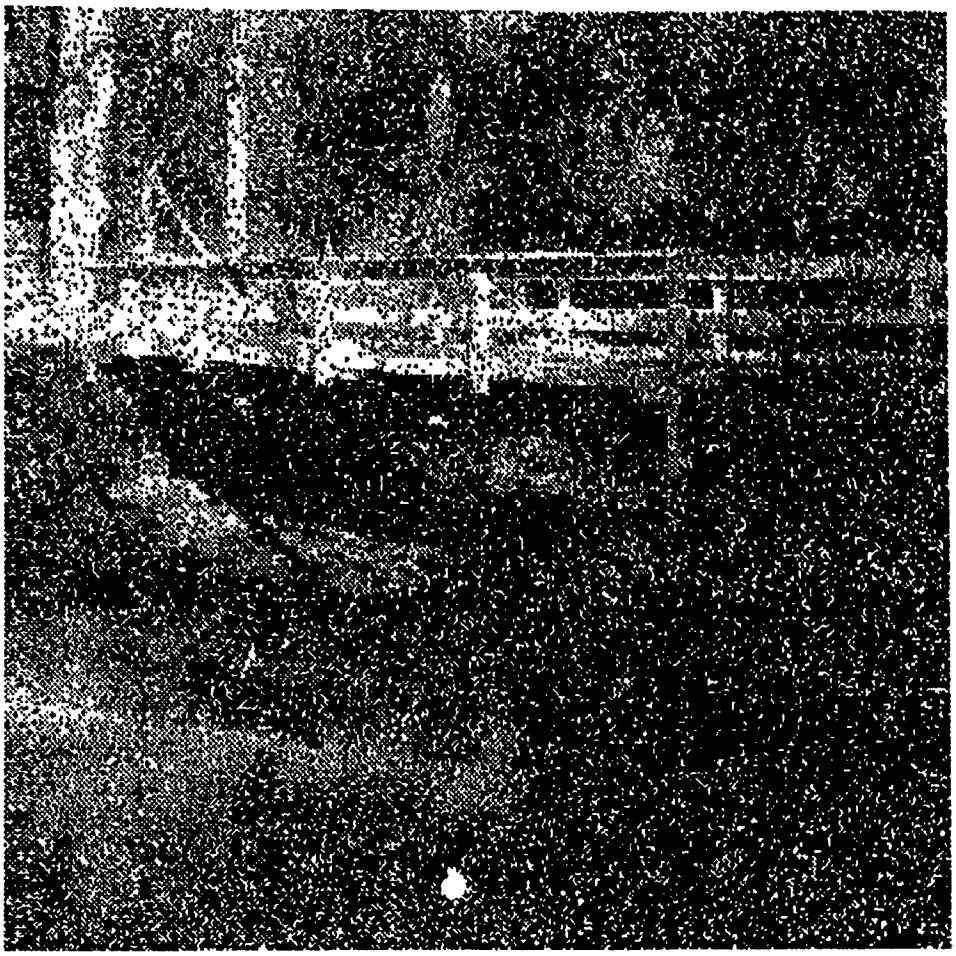
Two test images, each corrupted with a different type of impulsive noise, are considered. The performance of the optimal filters is specified in terms of the space average of the absolute error between the pixel values of the uncorrupted image and those of the corrupted or estimated version of the same. Inspite of being a space average, it is common practice to denote this performance measure by MAE. In each experiment, the samples in the upper left quarter of the original image and its corrupted version have been used for the training of the LSSL and LSS filter architecture.

Experiment I. In this experiment, the problem of finding an LMAE WOS filter of window size 3×3, for filtering a test image which is corrupted by both positive and negative impulsive noise is considered. Figure 4.4(a) shows the original Bridge image used in the experiment. The size of the image is 512×512, and the number of grey levels is 64. The image in Figure 4.4(b) is a 22% corrupted version of the original Bridge image in Figure 4.4(a). The noise is additive and uniformly distributed in the interval $[-22, 44]$, and the saturation at 0 (black) and 63 (white) gives 5% black-level impulsive noise and 11% white-level impulsive noise. The mean absolute error in this corrupted image is MAE=6.950352. A value of $\mu = 0.004$ is used in the implementation of LMAE algorithms. The samples in the upper left quarter of the images shown in Figures 4.4(a) and (b) have been used for training the LSSL and LSS filter architectures.

The results of filtering the image of Figure 4.4(b) using the optimal LSSL and LSS architectures, are shown in Figures 4.4(c) and (d), respectively. The MAE in

(a)

Figure 4.4: Filtering a test image of size 512 × 512 with 64 grey levels that is corrupted by both positive and negative impulsive noise, using 3 × 3 filters. (a) Original Bridge image. *(Continued on next page.)*

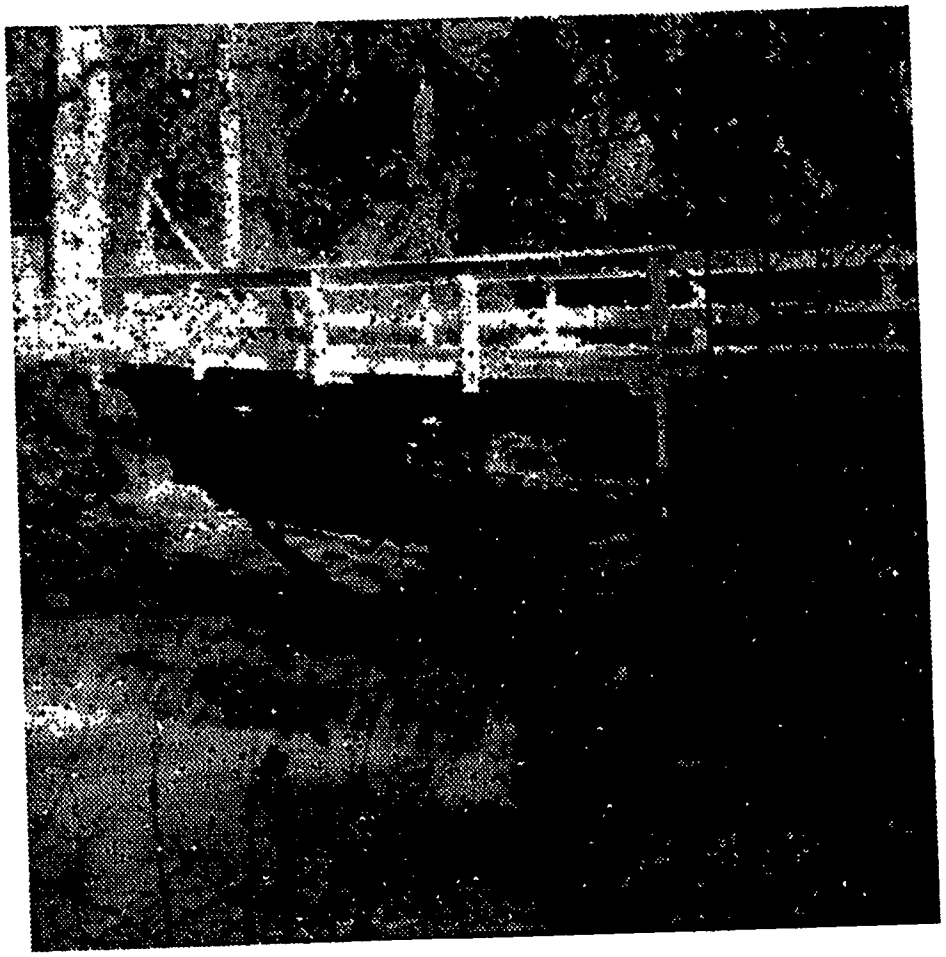


(b)

Figure 4.4: *(Continued.)* (b) A 22% (MAE=6.950352) corrupted version of Bridge image. The noise is additive and uniformly distributed in the interval $[-22, 44]$. The saturation at 0 (black) and 63 (white) gives a 5% black-level impulsive noise and a 11% white-level impulsive noise. *(Continued on next page.)*

(c)

Figure 4.4: *(Continued.)* (c) The filtered image by applying the optimal LSSL architecture on the image shown in (b); MAE=2.294079. *(Continued on next page.)*



(d)

Figure 4.4: *(Continued.)* (d) The filtered image obtained by applying the optimal LSS filter architecture on the image shown in (b); MAE=2.432912.

the filtered image using the proposed design is 2.294079, whereas it is 2.432912 when the image is filtered using the LSS architecture. Thus, the former architecture gives an improvement of 6% over the latter one. The weight values of the optimal LSSL architecture are obtained as

$$\begin{bmatrix} 0.170512 & 0.098603 & 0.130514 \\ 0.142961 & -0.225842 & 0.149439 \\ 0.089436 & 0.058104 & 0.098706 \end{bmatrix}$$

$$w_T = 0.5\,,$$

whereas those of using the optimal LSS architecture are found to be

$$\begin{bmatrix} 0.130396 & 0.119378 & 0.092875 \\ 0.150569 & 0.280302 & 0.109578 \\ 0.043546 & 0.021386 & 0.030245 \end{bmatrix}$$

$$w_T = 0.656531\,.$$

Experiment II. In this experiment, the problem of finding an LMAE WOS filter of window size 3×3, for filtering a test image corrupted by positive impulsive noise is considered. Figure 4.5(a) shows the original Lenna image that has been used. The size of the image is 256×240, and the number of grey levels is 64. The noise is additive and uniformly distributed in the interval $[0, 63]$. Five different cases of noise-corruption have been considered. For each case, the design of a WOS filter is performed by employing the LSSL and LSS architectures with $\mu = 0.002$. The samples in the upper left quarter of the uncorrupted and corrupted images were used for training the LSSL and LSS filter architectures. The results are summarized in Table 4.1. It is seen that for higher levels of noise-corruption in the input image, the WOS filters designed by using the LSSL architecture yield increasingly better results than those obtained by using the LSS architecture. For instance, when the

Table 4.1: Comparison of the results of WOS filtering by using the LSSL and LSS architectures.

Mean Absolute Error (MAE)			*Percentage*
Input image	*Optimal LSS Filter*	*Optimal LSSL Filter*	*Improvement*
17.283	3.873	3.025	22
12.364	3.791	2.972	22
9.756	3.623	2.958	18
6.117	3.256	2.935	9
4.231	2.943	2.931	3

noise level in the input image is increased from MAE=4.231 to MAE=17.283, the filtering capability of the WOS filters using the LSSL architecture increases from 3% to 22% over the WOS filters using the LSS architecture.

Figure 4.5 shows the images illustrating the results of a WOS filtering of the Lenna image corrupted with an MAE=17.283 by using the optimal LSSL and LSS architectures. For this noise level, the weight values corresponding to the optimal LSSL architecture are

$$\begin{bmatrix} 0.172580 & -0.133225 & 0.105357 \\ 0.176109 & 0.206681 & -0.146193 \\ -0.031987 & -0.002175 & -0.039405 \end{bmatrix}$$

$$w_T = 0.5\,,$$

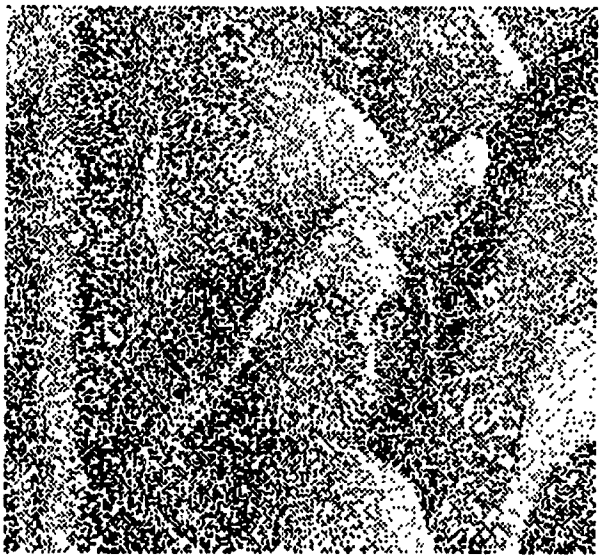
whereas those obtained from the LSS architecture design are

$$\begin{bmatrix} 0.223461 & 0.145476 & 0.151923 \\ 0.250592 & 0.266648 & 0.151680 \\ 0.086436 & 0.046997 & 0.055393 \end{bmatrix}$$

$$w_T = 1.060690\,.$$

If the designed LSSL architecture is converted to the corresponding LSS architecture, all the weights in the latter will be positive, with the same magnitude as those of

(a)



(b)

Figure 4.5: Filtering a test image of size 256×240 with 64 grey levels that is corrupted by positive impulsive noise, using 3×3 filters. (a) Original Lenna image. (b) A 65% (MAE=17.282747) corrupted version of Lenna image; the noise is additive and uniformly distributed in the interval $[0, 63]$, and the saturation at 63 (white) gives 47% white-level impulsive noise. *(Continued on next page.)*

(c)

(d)

Figure 4.5: *(Continued.)* (c) The filtered image by applying the optimal LSSL filter architecture on the image shown in (b); MAE=3.025345. (d) The filtered image by applying the optimal LSS filter architecture on the image shown in (b); MAE=3.873452.

the former. However, the w_T will be changed from 0.5 to 0.852985. Note that the weight and the threshold values in the converted LSS architecture are very different from those of the LSS architecture designed using the traditional approach [YAN93].

The above experiments show that by using the LSSL architecture, it is possible to achieve lower, or at least the same errors, as compared with those obtained with the LSS architecture, in which the weights are restricted to assume positive or zero values only.

4.6 Summary

In this chapter, an adaptive LMAE method for the design of WOS filters has been developed. It has been shown that the proposed design using the LSSL architecture yields a WOS filter that is approximately optimal in the MAE sense. The new design method is less constrained than the method given in [YAN93], while it still enjoys the implementation simplicity of stack filters. The implementation of the new adaptive LMAE design of WOS filters by using a binary-level LMS algorithm has also been discussed. In order to demonstrate the performance of the new design method, experimental results in image restoration from the observation data corrupted with impulsive noise have been provided. From these experiments, it has been seen that by using the new adaptive design method it is possible to achieve lower errors than those given by the method using the LSS architecture [YAN93]. As a final remark, we note that the work of this chapter can also be helpful in understanding the problem of finding the relation between stack filters and neural networks, that has been recently posed by Gabbouj, Coyle and Gallagher in [GCG92].

Chapter 5

Concluding Remarks

5.1 Conclusions

The primary contributions of this thesis have been the development of a new filter configuration called linearly separable stack-like (LSSL) architecture, and the derivation of an algorithm for the adaptive design of the class of WOS filters realized by using the LSSL architecture. It has been shown that this adaptive design is less constrained than the adaptive methods which have been used in the past that restrict the design to the domain of LSS filter architectures only. As a result, it has been shown that by using the new design, lower errors can be achieved.

The LSSL filter has been defined based on introducing a new concept of generalized threshold decomposition of a multilevel finite sequence. It has been shown that in a stack-like filter architecture, although the binary input signals do not satisfy the stacking property in the strict sense, the binary output signals do. This property of the LSSL architecture has been called generalized stacking property. Due to this property, the basics of the optimality theory developed by Coyle and Lin [CL88] for the class of stack filters, still remain valid in the case of LSSL filter architectures. It has been demonstrated that in the multilevel signal domain, an LSSL filter architecture performs the WOS filtering operation.

An adaptive algorithm for the design of WOS filters using the LSSL architecture has been derived. Since the filter weights in this architecture can assume any real values, the algorithm is less constrained than that in the case of LSS filter architecture. An implementation of the new design algorithm has been proposed, by constructing a binary-level LMS algorithm.

The proposed design and implementation has been applied to the problem of restoring images that are corrupted with impulsive noise. Simulation results have shown that the WOS filters designed with the new method provide better results compared with those obtained by using the WOS filters designed with the LSS architecture.

5.2 Scope For Future Investigation

It has been shown that the algorithm for the adaptive design of WOS filters within the framework of the LSSL architecture is less constrained than the one using the LSS architecture. A theoretical study of the convergence of this new adaptive algorithm can be carried out.

An investigation can be undertaken for finding a closed-form solution to the problem of the MMAE design of the WOS filters using the LSSL architectures.

It has also been observed that any unate Boolean function satisfies the generalized stacking property. Consequently, the concept of stack-like filter architecture might be extended to the more general case of unate Boolean functions. In turn, this could bring new insights for further research in the theory of stack filters.

Finally, it may be worthwhile to look into the possibility of developing neural network implementation for the LSSL architecture of WOS filters, proposed in this thesis.

References

[AHL92] N. Ansari, Y. Huang, and J. H. Lin. Adaptive stack filtering by LMS and Perceptron learning. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 4:57–60, March 1992.

[BHM83] A. C. Bovik, T. S. Huang, and D. C. Munson, Jr. A generalization of median filtering using linear combinations of order statistics. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 31(6):1342–1350, December 1983.

[CG89] E. J. Coyle and N. C. Gallagher, Jr. Stack filters and neural networks. *Proceedings of the International Symposium on Circuits and Systems*, pages 995–998, 1989.

[CL88] E. J. Coyle and J. H. Lin. Stack filters and the mean absolute error criterion. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(8):1244–1254, August 1988.

[FCG84] J. P. Fitch, E. J. Coyle, and N. C. Gallagher, Jr. Median filtering by threshold decomposition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-32(6):1183–1188, December 1984.

[FCG85] J. P. Fitch, E. J. Coyle, and N. C. Gallagher, Jr. Threshold decomposition of multidimensional ranked order operations. *IEEE Transactions on Circuits and Systems*, CAS-32(5):445–450, May 1985.

[Fit87] J. P. Fitch. Software and VLSI algorithms for generalized rank order filtering. *IEEE Trans. Circuits and Systems*, 34(5):553 559, May 1987.

[Gab89] M. Gabbouj. *Estimation and structural-based approach for the design of optimal stack filters.* PhD thesis, Purdue University, West Lafayette, IN, December 1989.

[GC90] M. Gabbouj and E. J. Coyle. Minimum mean absolute error stack filtering with structural constraints and goals. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(6):955–968, June 1990.

[GC91] M. Gabbouj and E. J. Coyle. On the LP which finds an MMAE stack filter. *IEEE Transactions on Signal Processing*, 39(11):2419 2424, November 1991.

[GCG92] M. Gabbouj, E. J. Coyle, and N. C. Gallagher, Jr. An overview of median and stack filtering. *Circuits, Systems and Signal Process.*, 11(1):7 45, January 1992.

[Gil54] E. N. Gilbert. Lattice theoretic properties of frontal switching functions. *Journal of Mathematical Physics*, 33:57–67, April 1954

[GW81] N. C. Gallagher, Jr. and G. L. Wise. A theoretical analysis of the properties of median filters. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(6):1136–1141, December 1981.

[HAN91] O. Y. Harja, J. T. Astola, and Y. A Neuvo. Analysis of the properties of median and weighted median filters using threshold logic and stack filter representation. *IEEE Transactions on Signal Processing*, 39(2):395 410, February 1991.

[Hay91] S. Haykin. *Adaptive Filter Theory.* Prentice-Hall, Englewood Cliffs, NJ, second edition, 1991.

[LB89] H. G. Longbotham and A. C. Bovik. Theory of order statistic filters and their relationship to linear FIR filters. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(2):275–287, February 1989.

[LC90] J. H. Lin and E. J. Coyle. Minimum mean absolute error estimation over the class of generalized stack filters. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(4):663–678, April 1990.

[LSC90] J. H. Lin, T. M. Sellke, and E. J. Coyle. Adaptive stack filtering under the mean absolute error criterion. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(6):938–954, June 1990.

[Mur71] S. Muroga. *Threshold Logic and Its Applications.* John Wiley & Sons, New York, 1971.

[PB88] F. Palmieri and C. G. Boncelet, Jr. A class of nonlinear adaptive filters. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 1483–1486, April 1988.

[Pee80] P. Z. Peebles, Jr. *Probability, Random Variables, and Random Signal Principles.* McGraw-Hill, New York, 1980.

[PS82] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

[Qui53] W. V. Quine. Two theorems about truth functions. *Boletin Sociedad Matematica*, Y:64–70, March 1953.

[She69] C. L. Sheng. *Threshold Logic.* Academic Press, New York, 1969.

[WCG86] P. D. Wendt, E. J. Coyle, and N. C. Gallagher, Jr. Stack filters. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(4):898–911, August 1986.

[WS85] B. Widrow and S. D. Stearns. *Adaptive Signal Processing.* Prentice-Hall, Englewood Cliffs, NJ, 1985.

[YAN92] L. Yin, J. Astola, and Y. Neuvo. Neural filters: a class of filters unifying FIR and median filters. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 4:53–56, March 1992.

[YAN93] L. Yin, J. T. Astola, and Y. A. Neuvo. Adaptive stack filtering with application to image processing. *IEEE Transactions on Signal Processing*, 41(1):162–184, January 1993.

[ZGN91] B. Zeng, M. Gabbouj, and Y. Neuvo. A unified design method for rank-order, stack, and generalized stack filters based on classical bayes decision. *IEEE Transactions on Circuits and Systems*, 38(9):1003–1020, September 1991.