

DESIGN OF OPTIMAL STACK FILTERS WITH STRUCTURAL CONSTRAINTS UNDER THE MAE CRITERION

Moncef Gabbouj and Edward J. Coyle
School of Electrical Engineering
Purdue University
West Lafayette, IN 47907

Abstract: In this paper, the theory of root signals for stack filters is combined with the theory of minimum mean absolute error stack filtering. This new, unified theory allows the designer to pick a filter which minimizes noise subject to constraints on its structural behavior.

1. Introduction

Median filters preserve signals, called *root signals*, whose structure consists of strings of monotonic "edges" and constant-valued regions [1,2]. They eliminate signal structures which are impulsive in nature since repeated filtering of a signal with a median filter will eventually reduce that signal to a root signal [1]. These two facts indicate that the median filter should be very good at removing impulsive noise from signals and images which are similar to root signals. This, as discussed for example in [1,2], has led to the use of median filters in many applications in digital image processing.

The goal of this paper is to extend the use of these concepts of preservation, removal and modification of specific signal structures to the set of filters known as stack filters [3]. Since their definition is motivated by the two properties of the median filter, namely the threshold decomposition and the stacking property; it should be possible to characterize stack filters by the signal structures they preserve or delete.

This paper shows that it is possible to design a stack filter which minimizes the mean absolute error subject to constraints on, or goals for, its behavior with regard to any list of signal structures specified by the designer. Thus, this approach unites the structural approach and the estimation approach to the design of stack filters under a single analytical methodology.

This paper is organized as follows. In Section 2, a general approach to the definition and properties of binary root signals of stack filters is reviewed. Specifically, this section leads to a characterization of binary root signals of stack filters as cycles in a directed graph. The theory of root signals is combined, in Section 3, with the estimation theory to produce tests for specific structural behavior and noise minimization. Section 4 contains the conclusion.

2. Root Signals for Stack Filters

Since stack filters obey the above two properties, it was shown that filtering an M -valued signal by a stack filter $S_f(\cdot)$ based on the positive Boolean function $f(\cdot)$ reduces to filtering each of its $(M-1)$ binary threshold signals by the binary stack filter $f(\cdot)$ and summing up their output.

Thus, to study the root signal behavior of a stack filter, it is best to begin with a study of the set of binary roots of the filter.

2.1. Stack Filter's Digraph

Suppose we wish to first determine whether a particular stack filter has any nontrivial root signals, and then, if it has roots, to specify them precisely. Both the existence and nature of the root signals for a stack filter can be determined with the help of the stack filter's digraph [4], the construction of which we now address.

The first step in this construction is the development of a digraph which specifies all possible sequences of binary

sequences that can be observed as a signal moves through a window of fixed width.

By motion of the signal through a filter window, we mean the legal transitions between sequences observed in the window. For instance, if a window of width three slides, one point at a time, from left to right over a signal, then $110 \rightarrow 101$ and $001 \rightarrow 010$ are legal transitions, whereas $110 \rightarrow 010$ and $101 \rightarrow 000$ are illegal transitions. The arrow under each sequence indicates the direction in which the filter window advances. The rightmost sample of each new state is the new sample that entered the window.

Define a graph called *window process digraph* [4] $D_n = (V, E)$ for a window of width n as follows: Define V to contain 2^n nodes, and label, without repeats, each node in V with one of the 2^n possible binary sequences of length n . The set of edges E is the set of all legal transitions defined as the filter's window moves from left to right over any signal. The window process digraph D_3 is shown in Fig. 2.1.

From the window process digraph D_n for a window of width n , we then construct the *stack filter's digraph* D_f for a particular stack filter $f(\cdot)$ with window width n . It is this graph which will determine whether the filter $f(\cdot)$ has roots or not, and which will allow the characteristics of any roots which exist to be determined.

Since $f(\cdot)$ is a binary stack filter of window width n , its output, which is binary, is well defined for each of the states

of the window digraph. Define $ref = \left\lfloor \frac{n}{2} \right\rfloor + 1$ and $ref(w_i)$ to be

the value of the ref sample of w_i from the left. If node j in the window process digraph is labeled with a state w_j with $ref(w_j)$ different from $f(w_j)$, delete this node from the window process digraph. Also delete any edges which were incident upon this node. If the filter output $f(w_j)$ coincides with $ref(w_j)$, then keep node j in the digraph. Carry out this procedure for each node j in the window process graph.

The graph which results from the above thinning of the window process graph is called the stack filter's digraph D_f for the stack filter $f(\cdot)$. Figure 2.2 shows the stack filter's digraph for the window width three median filter.

Definition 2.1: A sequence is a finite binary string. A signal is an infinite binary string. A sequence is said to be a root sequence of the window width n stack filter $f(\cdot)$ if all samples of its n -reduced version (i.e., the original sequence with $(ref-1)$ samples removed from both ends) are preserved under $f(\cdot)$. A signal is said to be a root signal of the stack filter $f(\cdot)$ if the signal is invariant to the filter $f(\cdot)$.

The following two Lemmas relate root sequences to paths in the filter's directed graph.

Lemma 1: Any walk in the stack filter's directed graph D_f corresponds to a sequence that is invariant to the filter $f(\cdot)$.

Lemma 2: Any invariant sequence corresponds to a walk in the filter's directed graph D_f .

The preceding two lemmas show the correspondence between root signals for a stack filter and that filter's digraph. One would then expect that the existence of nontrivial root signals -- invariant signals of infinite length -- would correspond to elementary cycles in the filter's digraph.

Theorem 1: A stack filter $f(\cdot)$ preserves nonconstant signals if and only if its directed graph D_f contains one or more of the following:

- a) A nontrivial elementary cycle.
- b) A path that starts with one of the trivial elementary cycles and ends with the other.

For a proof of Theorem 1, see [4].

Remarks: Some stack filters' digraphs contain disjoint cycles: that is, two or more cycles which are pairwise disjoint, i.e. they do not share any nodes. This will be particularly interesting when we consider cycle breaking and cycle preserving.

2.2. Disjoint Cycles in Digraphs

An interesting characteristic of directed graphs is that they, sometimes, contain disjoint elementary cycles. This will prove important in a stack filter design example provided later in this paper.

An important question one might ask is: "For what window width, if any, is there a stack filter whose directed graph contains disjoint cycles?" By enumerating all stack filters of window width less than or equal to four and investigating their corresponding directed graphs, we found the following result.

Theorem 2: No stack filter's directed graph of any stack filter of window width less than five contains disjoint cycles.

On the other hand, there exist stack filters of higher window widths whose digraphs contain completely disjoint cycles.

Example 2.1: The following is the Boolean expression of a stack filter of window width five which is the MMAE filter for a particular filtering problem.

$$f(x_1, x_2, x_3, x_4, x_5) = x_1 x_3 x_5 + x_1 x_4 + x_2 x_3 + x_2 x_5 + x_3 x_4.$$

The filter's digraph contains two cycles which are disjoint:

Cycle 1: 01010 \rightarrow 10101 \rightarrow 01010

Root signal: ... 0101010101010 ...

Cycle 2:

01100 \rightarrow 11000 \rightarrow 10001 \rightarrow 00011 \rightarrow 00110 \rightarrow 01100

Root signal: ... 0110001100011 ...

Different cycles can span the same set of nodes [4]. Hence, in order to discriminate between such cycles, we need to change the window width of the filter. The following theorem tells us which way to go.

Theorem 3: For any two signals corresponding to two elementary cycles in D_n which span the same set of nodes, there does not exist a stack filter of window width less than or equal to n that will preserve one signal and alter the other.

An algorithm to increase the window width of the filter in such a case is available in [4].

3. Designing Stack Filters

When the LP in [5] is used to find a stack filter which minimizes the MAE between its output and a desired input signal, it is possible that the resulting filter may possess structural behavior that is not desirable.

For instance, suppose that a filter is optimal for a certain noise reduction problem, but it possesses root signals which contain oscillatory structures. The designer may, however, consider preservation of such oscillatory structures to be undesirable structural behavior.

Alternatively, suppose it is known that the signal to be recovered from the noise has specific structural characteristics. It might, for instance be known that any

realization of the desired signal is one of the root signals of the median filter of a certain window width. It could thus be characterized in terms of sequences of structures such as edges and constant valued regions [1,2,6,7]. It would be reasonable in some situations, such as high SNR situations, to specify that these structures be root signals or root sequences of the stack filter.

Of course, the structural constraints or goals may consist of a list of different structures to be preserved, deleted or modified. The list might appear as follows:

- 1_ No oscillations of period 2 shall be preserved by $f(\cdot)$ ($f(\cdot)$ is our binary filter).
- 2_ The filter should not alter positive going edges in the signal.
- 3_ Negative going impulses are to be preserved.
- 4_ A specific set of sequences must be preserved by the filter.

To account for these structural constraints and goals, the LP in [5] which finds a stack filter which is best in the MAE sense must be modified. Two approaches are possible, either modify the objective function of the LP or append more constraints. The advantages and disadvantages of these two methods will be discussed as the methods are presented.

Throughout this section, all sequences and signals are binary unless otherwise specified. The extension to multi-level signals is carried out in [4].

3.1. Modifying the Constraints

3.1.1. Preserving Root Signals

In order to preserve any particular root signal, the elementary cycle (or the compound cycle) corresponding to this root signal must remain in the stack filter's digraph; that is, all nodes involved must be in V .

Suppose that (u_1, u_2, \dots, u_k) make up the cycle we wish to preserve. Since u_i belongs to V for $i=1, 2, \dots, k$, $P_f(1 | u_i)$ is known for $i=1, \dots, k$. It is assigned the value 0 if $ref(u_i) = 0$; otherwise, it is assigned the value 1. These k new constraints are then added to the set of constraints of the original LP problem formulated in [5].

Example 3.1: Suppose that, in addition to minimizing noise, a window width three stack filter $f(\cdot)$ is to preserve oscillations in the signal wherever they occur. This implies that we want $f(\cdot)$ to preserve the following signal: ... 010101 ... The corresponding elementary cycle is 010 \rightarrow 101 \rightarrow 010. Hence, the following constraints are appended to the LP: (a) $P_f(1 | 010) = 1$ and (b) $P_f(1 | 101) = 0$. These new constraints ensure that nodes 010 and 101 will appear in D_f and that oscillations of period 2 are preserved under $f(\cdot)$. Alternatively, we can group these 2 new constraints in one equation: (c) $(1 - P_f(1 | 010)) + P_f(1 | 101) = 0$. This equation is equivalent to equations (a) and (b) whenever the filter's decisions are hard decisions. This can be generalized to any k -node cycle. Thus, only one additional constraint per cycle is needed to preserve any set of cycles.

Alternatively, we can use the new constraints to reduce the size of the LP. Let $C = \{v_i : v_i \in Cycle\}$ and $K = \{i : v_i \in C\}$. Then $P_f(1 | v_i) = ref(v_i)$ for all $v_i \in C$. This in turn will determine the values of at least $|K|$ unknown probabilities and we are left with at most $2^n - |K|$ unknowns for which we must solve. Using the stacking property, more variables may be determined once these $|K|$ variables become known, e.g. suppose that $P_f(1 | 110)$ was set to be 1, then $P_f(1 | 111)$ must also be 1 since 110 stacks on top of 111. We next present a step by step algorithm to preserve cycle C with $|K|$ nodes.

Algorithm:

Step 1_ Set $P_f(1 | v_i) = ref(v_i)$ for all $v_i \in C$

Step 2_ If $P_f(1 | v_i) = 1$ then

for any v_j such that $v_j > v_i$, set $P_f(1 | v_j) = 1$

else

for any v_k such that $v_k < v_i$, set $P_f(1 | v_k) = 0$.

Step 3. Solve the reduced LP problem to compute the remaining unknowns.

□

3.1.2. Removing Cycles

As discussed previously, eliminating sequences or signals is equivalent to breaking the corresponding cycles in the stack filter's digraph. A cycle can be broken by deleting one or more nodes in that cycle.

Assume that v_i is, for some reason, the node to be removed from the cycle. Then we must have $P_f(1 | v_i) = 1 - \text{ref}(v_i)$. This additional constraint is added to the LP as in the previous subsection.

Note that choosing which node is to be removed is arbitrary and might or might not interfere with other features of the optimal filter (computed based only on the stacking constraints, with minimizing the MAE as the goal). Hence, it would be better if, somehow, this operation of cycle breaking could be introduced to the LP as an additional constraint in such a way that the LP is given the freedom to decide which of the nodes in the cycle is to be removed. The result will be a filter which breaks the cycle in those spots which contribute the most to minimizing the mean absolute error.

To illustrate how this can be done, suppose we have a k -node cycle v_1, v_2, \dots, v_k that we want to break. Somewhere along the path of the cycle an edge must be removed or, equivalently, an error must be made at one or more nodes (by error, we mean $f(v_i) \neq \text{ref}(v_i)$ for some $i=1, 2, \dots, k$). Consider the following examples.

Example 3.2: Suppose we wish to break the following window width three cycle: $100 \rightarrow 001 \rightarrow 010 \rightarrow 100$ in the digraph of the stack filter whose Boolean expression is given by $f(x_1, x_2, x_3) = x_2 + x_1 x_3$. Then, one or more of the following must hold:

- a) $P_f(1 | 010) = 0 \rightarrow P_f(0 | 010) = 1$
- b) $P_f(1 | 001) = 1 \rightarrow P_f(0 | 001) = 0$
- c) $P_f(1 | 100) = 1 \rightarrow P_f(0 | 100) = 0$

If we wish to break the cycle in all three nodes, we simply require that

$$P_f(0 | 010) + P_f(1 | 001) + P_f(1 | 100) = 3.$$

If we wish it to be broken only at one or more nodes, then we could require

$$P_f(0 | 010) + P_f(1 | 001) + P_f(1 | 100) \geq 1.$$

The addition of this constraint will break the cycle provided the linear program still has a solution in which each of the filter decision probabilities in the inequality is integer valued. If they are not integer valued, then the resulting filter is difficult to interpret since it randomizes its outputs. The filter may sometimes preserve the cycle; at other times it may delete it. In the MMAE filtering problem which yielded the filter above, the addition of this inequality constraint still results in an integer solution. The new optimal stack filter is $f(x_1, x_2, x_3) = x_1 + x_2$, which has no nontrivial roots.

If the cycle to be broken is completely disjoint from the rest of the digraph, the designer can get rid of it without affecting the rest of the graph. On the other hand, there are situations where the removal of one cycle causes the destruction of other cycles, as the following example shows.

Example 3.3: Recall the filter discussed in Example 2.1. The Boolean expression of the filter is:

$$f(x_1, x_2, x_3, x_4, x_5) = x_1 x_3 x_5 + x_1 x_4 + x_2 x_3 + x_2 x_5 + x_3 x_4.$$

The filter preserves (among other cycles) Cycle 1 and Cycle 2. In fact, D_f contains 13 elementary cycles. Here, we will try to break Cycle 1 first, then Cycle 2, and finally, Cycle 1 and Cycle 2.

Since Cycle 1 consists of oscillations, it will be eliminated by requiring that

$$(a) P_f(1 | 01010) + P_f(0 | 10101) \geq 1.$$

The new optimal filter has the following Boolean expression:

$$f(x_1, x_2, x_3, x_4, x_5) = x_1 x_4 + x_2 x_3 + x_2 x_5 + x_3 x_4.$$

The new D_f contains 12 cycles, i.e. only Cycle 1 has been removed.

To break Cycle 2, the following must hold:

$$(b) P_f(1 | 00011) + P_f(0 | 00110) + P_f(0 | 01100) + P_f(1 | 10001) + P_f(1 | 11000) \geq 1.$$

The new optimal filter has the following Boolean expression:

$$f(x_1, x_2, x_3, x_4, x_5) = x_1 x_3 + x_1 x_4 + x_2 x_5 + x_3 x_4.$$

The new D_f contains ten cycles. Two extra cycles were destroyed as a result.

Finally, to break Cycle 1 and Cycle 2, both (a) and (b) must hold. The new optimal filter has the following expression:

$$f(x_1, x_2, x_3, x_4, x_5) = x_1 x_3 + x_1 x_4 + x_2 x_4 + x_2 x_5 + x_3 x_4.$$

The new stack filter's digraph of $f(\cdot)$ contains only nine cycles. The above operation caused two extra cycles to be removed. It is interesting to note that in all the cases considered above, the LP with the additional constraint still yielded an integer solution. This will be discussed in more detail later.

3.1.3. Simultaneously Removing and Preserving Cycles; Randomization

Designing stack filters to preserve certain cycles or to remove certain cycles by modifying the constraints is done as described in Sections 3.1.1 and 3.1.2, respectively. However, when we attempt to design a dual-purpose stack filter, which is a stack filter designed to preserve some cycles and to not preserve others, it is possible that the resulting set of constraints may be infeasible.

Proposition 3.1: If the additional constraints resulting from preserving cycles and removing cycles make the set of constraints infeasible, then there exists no stack filter with the prescribed structural behavior for that particular window width.

One case of interest which yields infeasible constraints was discussed in [4]. It is the case in which one cycle to be preserved and one to be removed both span the same set of nodes. Although this is a disadvantage of this design procedure, it is an important result concerning the filter window width. Theorem 3 tells us that a higher window width is required in order to accomplish the desired task.

A serious disadvantage of the above approach of adding constraints is the fact that the resulting LP may no longer have an integer solution. Previously, before adding any structural constraints, integral solution to the LP was guaranteed by the fact that the constraint matrix was originally totally unimodular (TUM) [8]. The addition of new constraints may distort the constraint matrix and may result in randomization of one or more variables at the output of the resulting optimal filter.

As can be seen from the examples considered in the previous subsections, though, it appears that the problem of randomization does not always arise. Indeed, we had to purposely search for cases in which randomization was necessary before we found one. Determining when randomization is required and when it is not is an interesting open problem in this research area.

Furthermore, even if the optimal filter must randomize, one of the deterministic filters among the set over which it randomizes can be used. Although the resulting filter is not optimal, it will be nearly optimal, and probably acceptable.

3.2. Modifying the Objective

In the previous subsection, it was shown how a stack filter which is optimal for noise reduction while satisfying constraints on its structural behavior could be obtained by adding these constraints to the original LP used to find a MMAE stack filter. It was noted, however, that this method

may sometimes cause the optimal solution to be a filter which randomizes some of its output decisions. If this is the case, or if none of the filters over which the randomization takes place is acceptable, an alternative approach can be taken.

This alternative method is based on an iterative process which may take more time to perform, but which, if successful, ensures that the filter which is finally chosen is a deterministic filter.

This process consists of changing the cost coefficients associated with certain decision variables in the objective function of the linear program in a systematic way until the desired output is obtained. Before making any changes in the objective, we run the LP with the current objective and check the resulting optimal filter it yields to determine if it has the desired structural behavior. This is accomplished by examining the filter's digraph to determine if it has the cycles which are desired and does not have the cycles which are not desired.

If it does not have the desired behavior, the next step is to perform a cost ranging analysis [9] on each cost coefficient associated with every node in the desired and undesired cycle(s). Cost ranging is usually used in sensitivity analysis to determine the range within which each cost coefficient is allowed to vary without changing the optimal point, but in this case it tells us the minimal change to a cost coefficient that will assure a change in the optimal filter.

Now, suppose a certain cycle is to be preserved and that one of its nodes is not in D_f . The designer will attempt to vary the cost coefficient, associated with that decision variable, so that this node will appear in the new D_f . Cost ranging will provide the minimal amount by which that cost coefficient must be updated to make the desired change. Similarly, if a certain cycle is to be removed from D_f , a cost ranging will be performed on each node (actually, on each cost coefficient associated with each node) in the cycle, and the optimal solution with the minimal objective will be selected.

By preserving the structure of the constraint matrix, the above method overcomes the worst drawback of the previous design procedure by ensuring that the resulting filter does not randomize. This method might require, however, a lengthy iterative process which involves scores of tedious computations. Therefore, it is not recommended and no examples using this method are provided. A simpler alternative to designing a filter with the desired structural behavior by modifying the objective function can be found in [10].

4. Conclusion

In this paper, we combined two filtering approaches: a root signal approach and MAE estimation approach. In Section 2, we reviewed a means to extract root signals for any stack filter by manipulating its directed graph. In Section 3, we presented two alternatives for designing stack filters which will preserve and/or remove certain signals. The first method modifies the constraints of the LP while the second method alters the objective without changing the constraints.

The extension of this paper to two dimensions can be found in [10].

References

- [1] N.C. Gallagher, Jr., and G.L. Wise, "A theoretical analysis of the properties of median filters," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-29, no. 6, Dec. 1981.
- [2] S.G. Tyan, "Median filtering: Deterministic properties," *Two-dimensional digital signal processing II: Transforms and median filters*. T.S. Huang, Ed. New York: Springer-Verlag, 1981.
- [3] P.D. Wendt, E.J. Coyle, and N.C. Gallagher, Jr., "Stack filters," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-34, no. 4, pp. 898-911, Aug. 1986.

- [4] M. Gabbouj and E.J. Coyle, "MMAE Stack Filtering with Structural Constraints and Goals," *IEEE Trans. on Acoustics, Speech and Signal Processing*, to appear.
- [5] E.J. Coyle and J.-H. Lin, "Stack filters and the mean absolute error criterion," *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-36, no. 8, Aug. 1988.
- [6] D.H. Yom and S. Ann, "Directed graph representation for root-signal set of median filters," *Proceedings of the IEEE*, vol. 75, no. 11, Nov. 1987.
- [7] P.D. Wendt, E.J. Coyle, and N.C. Gallagher, Jr., "Some convergence properties of median filters," *IEEE Trans. on Circuits And Systems*, vol. CAS-33, no. 3, pp. 276-286, March 1986.
- [8] C.H. Papadimitriou, and K. Steiglitz, *Combinatorial optimization: algorithms and complexity*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.
- [9] L. Cooper and D. Steinberg, *Methods and Applications of Linear Programming*, W. B. Saunders Co., Philadelphia, 1974.
- [10] E.J. Coyle, J.-H. Lin, and M. Gabbouj, "Optimal stack filtering and the estimation and structural approaches to image processing," *IEEE Trans. on Acoustics, Speech and Signal Processing*, to appear in December 1989 issue.

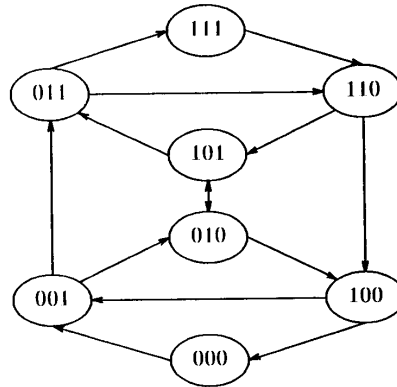


Figure 2.1 Window width three window process digraph.

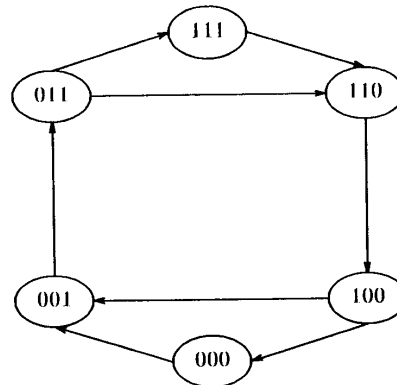


Figure 2.2 Stack filter digraph for the window width three median filter.