Iterative reweighted l_1 design of sparse FIR filters

Cristian Rusu[*], Bogdan Dumitrescu[**]

Abstract

Designing sparse 1D and 2D filters has been the object of research in recent years due mainly to the developments in the field of sparse representations. The main goal is to reduce the implementation complexity of a filter while keeping as much of the performance as possible. This paper describes a new method for designing sparse filters in the minimax sense using a mixture of reweighted l_1 minimization and greedy iterations. The combination proves to be quite efficient; after the reweighted l_1 minimization stage is used to introduce zero coefficients in bulk, a small number of greedy iterations serve to eliminate a few extra coefficients. Experimental results and a comparison with the latest methods show that the proposed method performs very well both in the running speed and in the quality of the solutions obtained.

Keywords: sparse filters, reweighted l_1 minimization, greedy algorithms.

1. Introduction

When designing 1D and 2D discrete-time filters, two important considered factors are: the filter quality and the cost of the implementation. In the case of 1D filters, the most important techniques that trade off these two issues are: restricting the coefficients of the filters to binary values [1] or powers of two [2] in order to completely remove the multiplications or replace them with shifts, recursive running-sum prefilters [3], cyclotomic polynomial prefilters

[*]C. Rusu is with the Department of Automatic Control and Computers, "Politehnica" University of Bucharest, Spl. Independenţei 313, Bucharest 060042, Romania (e-mail: cristian.rusu@schur.pub.ro).

[**]B. Dumitrescu is with the Tampere International Center for Signal Processing, Tampere University of Technology, 33101 Tampere, Finland, and Department of Automatic Control and Computers, "Politehnica" University of Bucharest, Bucharest 060042, Romania (e-mail: bogdand@cs.tut.fi).

[4], interpolated FIR filters [5] and frequency-response masking [6]. Many of these techniques can be extended to the 2D case, like for example frequency-response masking [7].

The approach in this paper is to use tools from the field of sparse representations [8] and the basic idea is to reduce the number of additions and multiplications by setting coefficients of the filters to zero. We call a filter sparse if it contains a relatively large number of zero coefficients. Since the problem of minimizing the number of nonzero coefficients is in general NP-hard, fast methods that give good approximations of the real solution were developed. The two most popular approaches that are used to induce sparsity are: convex relaxation to l_1 minimization [9] and greedy methods [10]. In this sense, previous work for sparse filter design includes the use of the orthogonal matching pursuit algorithm [11], greedy coefficient elimination [12] and linear programming [12], [13], [14]. In this paper we consider a combination of the two approaches to design filters in the minimax sense. In the first stage, reweighted l_1 minimization (IRL1) [15] is used to induce a relatively large number of zero coefficients and then, in the second stage, greedy iterations are used to cut down extra coefficients one by one. This combination proves to be very efficient both in the running time and the quality of the obtained solutions.

The article is structured as follows: Section 2 details the design problems for 1D and 2D FIR filters, Section 3 presents the proposed method called IRL1G and finally Section 4 outlines the results and a comparison with two other recent techniques used to design sparse filters.

2. Design problem formulation

2.1. 1D FIR filter design

Consider $H(z)$ the linear phase FIR filter with odd length N whose amplitude response is

$$A(\omega) = \sum_{k=-n}^{n} h_k e^{-kj\omega} = h_0 + \sum_{k=1}^{n} 2h_k \cos(k\omega), \tag{1}$$

where $h_k \in \mathbb{R}$, $n = (N-1)/2$ and the rightmost form of (1) results from the symmetry of the coefficients: $h_k = h_{-k}$. Given a desired response $D(\omega)$, the goal is to compute coefficients h_k such that the maximum absolute value of the error $E(\omega) = A(\omega) - D(\omega)$ is minimized over $0 \leq \omega \leq \pi$. Using a uniformly distributed dense grid of points $\omega_i \in [0,\pi], i = 1,\dots,L$ (transition bands not included), this problem can be formulated as the Linear Program (LP)

$$\begin{aligned} \underset{h_k,\delta}{\text{minimize}} \quad & \delta \\ \text{subject to} \quad & |E(\omega_i)| \le \delta. \end{aligned} \tag{2}$$

Plugging in the error expression and expanding the absolute value term we get the standard LP problem

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \boldsymbol{c}^{\boldsymbol{T}}\boldsymbol{x} \\ \text{subject to} \quad & \boldsymbol{Q}\boldsymbol{x} \le \boldsymbol{b}, \end{aligned} \tag{3}$$

where

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{C} & -\boldsymbol{v} \\ -\boldsymbol{C} & -\boldsymbol{v} \end{bmatrix} \in \mathbb{R}^{2L\times(n+2)}, \quad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{d} \\ -\boldsymbol{d} \end{bmatrix} \in \mathbb{R}^{2L}$$

$$\boldsymbol{C} = \begin{bmatrix} 1 & \cos(\omega_1) & \dots & \cos(n\omega_1) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \cos(\omega_i) & \dots & \cos(n\omega_i) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \cos(\omega_L) & \dots & \cos(n\omega_L) \end{bmatrix} \in \mathbb{R}^{L\times(n+1)}$$

$$\boldsymbol{v} = \begin{bmatrix} 1 \\ \vdots \\ 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbb{R}^{L}, \quad \boldsymbol{d} = \begin{bmatrix} D(\omega_1) \\ \vdots \\ D(\omega_i) \\ \vdots \\ D(\omega_L) \end{bmatrix} \in \mathbb{R}^{L}$$

$$\begin{aligned} \boldsymbol{x} &= [\begin{matrix} h_0 & 2h_1 & \dots & 2h_n & \delta \end{matrix}]^T \in \mathbb{R}^{n+2} \\ \boldsymbol{c} &= [\begin{matrix} 0 & 0 & \dots & 0 & 1 \end{matrix}]^T \in \mathbb{R}^{n+2}. \end{aligned}$$

This optimization problem generates the filter $H(z)$ whose coefficients are $h_{-n},\dots,h_0,\dots,h_n$ and is the closest in the minimax sense to the desired response $D(\omega)$.

2.2. 2D FIR filter design

Consider now $\tilde{H}(z_1, z_2)$ the linear phase FIR filter with N^2 coefficients; its transfer function is

$$\tilde{H}(z_1, z_2) = \sum_{k_1=-n}^{n} \sum_{k_2=-n}^{n} h_{k_1,k_2} z^{-k_1} z^{-k_2} = \boldsymbol{z}_1^T \tilde{\boldsymbol{H}} \boldsymbol{z}_2, \tag{4}$$

where $n = (N-1)/2$, $h_{k_1,k_2} \in \mathbb{R}$, $\boldsymbol{z}_1 = [z_1^n \ z_1^{n-1} \ \dots \ z_1^{-n+1} \ z_1^{-n}]^T$, $\boldsymbol{z}_2 = [z_2^n \ z_2^{n-1} \ \dots \ z_2^{-n+1} \ z_2^{-n}]^T$ and $\tilde{\boldsymbol{H}}$ is the matrix of the filter coefficients. The filter is quadrantally symmetrical and the coefficients obey to $h_{k_1,k_2} = h_{k_1,-k_2}$ and $h_{k_1,k_2} = h_{-k_1,k_2}$. So, the matrix $\tilde{\boldsymbol{H}}$ can be written as

$$\tilde{\boldsymbol{H}} = \begin{bmatrix} \tilde{\boldsymbol{H}}_{11} & \tilde{\boldsymbol{h}}_{12} & \tilde{\boldsymbol{H}}_{13} \\ \tilde{\boldsymbol{h}}_{21}^T & \tilde{h}_{22} & \tilde{\boldsymbol{h}}_{23}^T \\ \tilde{\boldsymbol{H}}_{31} & \tilde{\boldsymbol{h}}_{32} & \tilde{\boldsymbol{H}}_{33} \end{bmatrix}, \tag{5}$$

and the following hold: $\tilde{\boldsymbol{H}}_{11} = \text{flipud}(\text{fliplr}(\tilde{\boldsymbol{H}}_{33}))$, $\tilde{\boldsymbol{H}}_{13} = \text{flipud}(\tilde{\boldsymbol{H}}_{33})$, $\tilde{\boldsymbol{H}}_{31} = \text{fliplr}(\tilde{\boldsymbol{H}}_{33})$, $\tilde{\boldsymbol{h}}_{12} = \text{flipud}(\tilde{\boldsymbol{h}}_{32})$, $\tilde{\boldsymbol{h}}_{21}^T = \text{fliplr}(\tilde{\boldsymbol{h}}_{23}^T)$, where the function fliplr flips a matrix from left to right and the function flipud flips a matrix upside down. So, the distinct filter coefficients can be grouped in the matrix

$$\hat{\boldsymbol{H}} = \begin{bmatrix} \tilde{h}_{22} & 2\tilde{\boldsymbol{h}}_{23}^T \\ 2\tilde{\boldsymbol{h}}_{32} & 4\tilde{\boldsymbol{H}}_{33} \end{bmatrix}, \tag{6}$$

where $\tilde{\boldsymbol{H}}_{33} \in \mathbb{R}^{n^2}$, $\tilde{\boldsymbol{h}}_{23}^T \in \mathbb{R}^n$, $\tilde{\boldsymbol{h}}_{32} \in \mathbb{R}^n$ and $\tilde{h}_{22} \in \mathbb{R}$. Due to the symmetry of the coefficients, the amplitude response is

$$\begin{gathered} A(\boldsymbol{\omega}) = A(\omega_1, \omega_2) = h_{0,0} + \sum_{k_2=1}^{n} 2h_{0,k_2} cos(k_2\omega_2) + \\ \sum_{k_1=1}^{n} 2h_{k_1,0} cos(k_1\omega_1) + \sum_{k_1=1}^{n} \sum_{k_2=1}^{n} 4h_{k_1,k_2} cos(k_1\omega_1) cos(k_2\omega_2) = \boldsymbol{c}_{\boldsymbol{w}}^T \boldsymbol{h}, \end{gathered} \tag{7}$$

where

$$\begin{aligned} \boldsymbol{h} = \ & [h_{0,0} \quad 2h_{0,1} \quad \dots \quad 2h_{0,n} \quad 2h_{1,0} \quad \dots \quad 2h_{n,0} \quad 4h_{1,1} \quad \dots \quad 4h_{1,n} \quad \dots \quad 4h_{n,n}]^T \\ \boldsymbol{c}_{\boldsymbol{w}} = \ & [1 \quad \cos(\omega_2) \quad \dots \quad \cos(n\omega_2) \quad \cos(\omega_1) \quad \dots \quad \cos(n\omega_1) \\ & \cos(\omega_1)\cos(\omega_2) \quad \dots \quad \cos(\omega_1)\cos(n\omega_2) \quad \dots \quad \cos(n\omega_1)\cos(n\omega_2)]^T. \end{aligned} \tag{8}$$

Note that the vector $\boldsymbol{h}$ contains the elements of $\tilde{\boldsymbol{H}}$ from (6). Like in the 1D case, given a desired response $D(\boldsymbol{\omega})$, the goal is to compute coefficients h_{k_1,k_2} such that the maximum absolute value of the error $E(\boldsymbol{\omega}) = A(\boldsymbol{\omega}) - D(\boldsymbol{\omega})$ is minimized, where this time $\boldsymbol{\omega} = (\omega_1, \omega_2)$ and $\omega_1, \omega_2 \in [0, \pi]$. In the 2D case, each discretization point $\boldsymbol{\omega_i}$ has two components $(\omega_{i,1}, \omega_{i,2})$. Let $\tilde{n} = (n+1)^2$ be the number of distinct coefficients, then the reduced optimization problem can be formulated exactly like in the 1D case

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \boldsymbol{c}^T \boldsymbol{x} \\ \text{subject to} \quad & \boldsymbol{Q}\boldsymbol{x} \leq \boldsymbol{b}, \end{aligned} \tag{9}$$

where

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{C} & -\boldsymbol{v} \\ -\boldsymbol{C} & -\boldsymbol{v} \end{bmatrix} \in \mathbb{R}^{2L \times \tilde{n}+1}, \quad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{d} \\ -\boldsymbol{d} \end{bmatrix} \in \mathbb{R}^{2L}$$

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{c}_{\boldsymbol{\omega_1}}^T \\ \vdots \\ \boldsymbol{c}_{\boldsymbol{\omega_i}}^T \\ \vdots \\ \boldsymbol{c}_{\boldsymbol{\omega_L}}^T \end{bmatrix} \in \mathbb{R}^{L \times \tilde{n}}, \quad \boldsymbol{v} = \begin{bmatrix} 1 \\ \vdots \\ 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbb{R}^{L}, \quad \boldsymbol{d} = \begin{bmatrix} D(\boldsymbol{\omega_1}) \\ \vdots \\ D(\boldsymbol{\omega_i}) \\ \vdots \\ D(\boldsymbol{\omega_L}) \end{bmatrix} \in \mathbb{R}^{L}$$

$$\begin{aligned} \boldsymbol{x} &= [\quad \boldsymbol{h}^T \quad \delta \quad]^T \in \mathbb{R}^{\tilde{n}+1} \\ \boldsymbol{c} &= [\quad \boldsymbol{0}_{1\times\tilde{n}} \quad 1 \quad]^T \in \mathbb{R}^{\tilde{n}+1}. \end{aligned}$$

The order of the coefficients h_{k_1,k_2} in the solution $\boldsymbol{x}$ is determined by (8).

3. Design of sparse filters

Since the problems of designing sparse filters were formulated in the same manner for the 1D and 2D cases, this section describes an optimization method that works in both situations. In order to find high order filters that have a large number of zero coefficients and still manage to keep the error below a fixed value δ^{imposed}, we use a combination of the two basic approaches in the sparse representations field: reweighted l_1 minimization and greedy algorithms. The general structure of the proposed algorithm, called IRL1G, is described in Table 1.

Table 1: General overview of the IRL1G Algorithm

Input:	N - the length of the filter ω_i - the discretization points, $i = 1, \ldots, L$ $D(\omega_i)$ - desired response, $i = 1, \ldots, L$ δ^{imposed} - the maximum allowed error
Stage A:	IRL1 minimization detailed in Table 2
Stage B:	greedy iterations detailed in Table 3
Output:	the sparse filter H

In Stage A of the proposed method, the IRL1 minimization problem

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \boldsymbol{c}^T \boldsymbol{x} + \mu ||\boldsymbol{W}\boldsymbol{x}||_1 \\ \text{subject to} \quad & \boldsymbol{Q}\boldsymbol{x} \leq \boldsymbol{b} \\ & x_i = 0, \ \forall i \in S \\ & \boldsymbol{c}^T \boldsymbol{x} \leq \delta^{\text{imposed}} \end{aligned} \tag{10}$$

is solved; the l_1 penalization term is added to the criterion of (3) or (9) in order to promote sparse solutions. The diagonal matrix $\boldsymbol{W}$ contains the weights associated with each coefficient and is initially

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & 0 \end{bmatrix}, \tag{11}$$

i.e. the coefficients are equally weighted; the last diagonal zero corresponds to the error term. The weights are recomputed after each iteration. The idea is to put high weights for coefficients that have small absolute value in order to push them to zero and small weights for the large coefficients in absolute value because it is unlikely that those coefficients can be pushed to zero. The set S contains the indices of the coefficients that are forced to zero and so are eliminated from the next iterations. The set can grow at each iteration by the hard thresholding step (14). This means that the size of the optimization problem solved shrinks at each iteration, speeding up the running time of the overall algorithm. If the change in the solution x is below ϵ^{stop} from one iteration to the next, as checked in (12), then the procedure is terminated because further iterations will not make any progress. In the case of the 2D filters, we have modified the usual reweighting rule in (13) because there exists an extra incentive to nullify coefficients that belong to the $\tilde{\boldsymbol{H}}_{33}$ matrix: these coefficients participate four times in the final matrix $\tilde{\boldsymbol{H}}$, as opposed to only two times for the others (and once for the central coefficient). The extra weight is contained in the parameter $a > 1$.

Stage B consists of polishing the optimal filter with the support set S given by Stage A, which is computed by (15), using a few greedy iterations. The idea is to try to further eliminate the coefficients of the solutions that are the smallest in absolute value while keeping the optimization problem (15) feasible. This stage applies exactly the smallest coefficient rule (SCR) presented in [12].

Stage A proves to be far more efficient than Stage B, because in Stage A, in just a few iteration, a relatively large number of coefficients are set to zero, whereas in Stage B only one coefficient is set to zero for each LP solved.

4. Results

4.1. 1D case

We present here the results obtained by the IRL1G method for designing lowpass 1D filters with $\omega_p = 0.26\pi$ and $\omega_s = 0.34\pi$. The parameters of the algorithm are: maxSteps $= 15$, $\mu = 1$, $\epsilon = 10^{-6}$, $\epsilon^{\text{stop}} = 10^{-4}$, $\epsilon^{\text{cut}} = 10^{-7}$ and $L = 15N$. For comparison, the smallest coefficient rule (SCR) algorithm from [12] was also implemented. If the solution of the optimization problem ends up having M zero coefficients, then using this strategy, $M + 1$ LPs are solved. Compared to the method described in this paper that usually solves $8-9$ LPs to reach a solution, the SCR runs for a large number of iterations if M is large, making it harder to use in these settings. It is worth to mention that the IRL1 steps have the same basic idea: try to push to zero coefficients that have a small magnitude. The big difference is that while SCR can handle only one coefficient at a time, IRL1 can generate more zero coefficients in a single step.

Tests are conducted using filters of lengths from $N = 71$ to $N = 101$. For each filter, 12 simulations are conducted using uniformly distributed values of $\delta^{\text{imposed}} \in [1.5\delta^{\text{minimax}}, 2.5\delta^{\text{minimax}}]$ where δ^{minimax} is the minimax error obtained with the optimized full filter. After running 192 simulations, in 175 of the situations (91.16%), the two strategies performed the same. This means that the sparsity of the final solutions is the same, although the

Table 2: Stage A: Iterative reweighted l_1 minimization

Parameters

1. maxSteps > 1, the maximum number of iterations for the IRL1 minimization
2. $\epsilon, 0 < \epsilon \ll 1$, parameter used for the computation of weights
3. $\epsilon^{\text{stop}}, 0 < \epsilon^{\text{stop}} \ll 1$, used as an extra stopping condition
4. $\epsilon^{\text{cut}}, 0 < \epsilon^{\text{cut}} \ll 1$, elimination threshold used to decide if a filter coefficient is permanently set to zero
5. $a > 1$, extra penalizing weight

Procedure

```
for step = 1 ... maxSteps do
  1. solve problem (10)
  2. if solution $\boldsymbol{x}$ is feasible
       if $||\boldsymbol{x} - \boldsymbol{x}^{\text{previous}}||_2 < \epsilon^{\text{stop}}$ then break for loop      (12)
  3. if solution $\boldsymbol{x}$ is not feasible
       • return to previous solution $x = x^{\text{previous}}$ and set $S = S^{\text{previous}}$
       • decrease elimination threshold $\epsilon^{\text{cut}} = \epsilon^{\text{cut}}/10$
  4. compute new weights
       $W_{ii} = \begin{cases} \frac{a}{|x_i|+\epsilon}, & \text{2D case, if } x_i \in \tilde{\boldsymbol{H}}_{33} \\ \frac{1}{|x_i|+\epsilon}, & \text{otherwise} \end{cases}$      (13)
  5. decide the support of $\boldsymbol{x}$
       $S = \{i \mid |x_i| \leq \epsilon^{\text{cut}}\}$      (14)
```

Table 3: Stage B: Greedy iterations

1. solve the LP

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \boldsymbol{c}^T\boldsymbol{x} \\ \text{subject to} \quad & \boldsymbol{Q}\boldsymbol{x} \le \boldsymbol{b} \\ & x_i = 0,\ \forall i \in S \\ & \boldsymbol{c}^T\boldsymbol{x} \le \delta^{\text{imposed}} \end{aligned} \tag{15}$$

2. repeat until (15) is not feasible
   - add to S the index for which $\boldsymbol{x}$ has the smallest nonzero absolute value

     $$S = S \cup \{i \mid \underset{i}{\text{argmin}}|x_i|, \forall i \notin S\} \tag{16}$$

   - solve (15) with the new set S

solutions have a different support in some cases. In 5.20% of the cases, IRL1G performed better and in the remaining 3.64% the straight SCR algorithm reached a sparser solution. It is worth to mention that in both cases in which one approach outperformed the other, the difference was one extra zero coefficient.

Regarding the speed of the optimization procedures, our method takes approximately 25 seconds to complete on a filter of length $N = 101$, while SCR takes about 115 seconds. Simulations ran on an Intel i3 2.13GHz processor with 4GB of RAM.

4.2. 2D case

For the 2D case, simulations are done on four circular shaped lowpass filters with $\omega_p = 0.5\pi$ and $\omega_s = 0.7\pi$ and four diamond shaped lowpass filters with $\omega_p = 0.6\pi$ and $\omega_s = \pi$, of sizes 11×11, 17×17, 23×23 and 29×29. The grid discretization step size is 0.025π, giving $L = 1120$ discretization points for the diamond shaped filters and $L = 1304$ for the circular shaped filters, in the passband and stopband regions. Aside of the IRL1G, we also propose a variation that provides better results: the IRL1G method is applied three

Table 4: Values of parameter μ

	diamond shaped		circular shaped	
N	μ_1	μ_2	μ_1	μ_2
29	$\mu_1 = 10^{-3}$	$\mu_2 = 10^{-3}$	$\mu_1 = 0.03$	$\mu_2 = 10^{-3}$
23	$\mu_1 = 10^{-3}$	$\mu_2 = 10^{-3}$	$\mu_1 = 0.03$	$\mu_2 = 10^{-3}$
17	$\mu_1 = 0.03$	$\mu_2 = 10^{-3}$	$\mu_1 = 0.01$	$\mu_2 = 10^{-3}$
11	$\mu_1 = 0.1$	$\mu_2 = 0.1$	$\mu_1 = 0.1$	$\mu_2 = 1$

times with increasing values for $\delta_i^{\text{imposed}}$ with $i = 1, \ldots, 3$, where $\delta_3^{\text{imposed}}$ is the original δ^{imposed} and the first two are situated between the minimum possible error that can be reached with a filter of the given size and δ^{imposed}. Simulations show that good choices for the first two errors are a quarter and a half of the defined distance; we call this approach IRL1G$\times$3. By applying this strategy, the less important coefficients are discarded early on, when it is clear they are not needed. The support set S is transferred from one instance of IRL1G to the next. The results are compared with the minimax hard thresholding approach for designing sparse 2D filter presented in [14]. In this paper, the authors propose solving the standard l_1 minimization problem (10) (without the extra constraint on δ^{imposed} ,$\boldsymbol{W}$ as (11) and $S = \emptyset$) and then setting the smallest coefficients in absolute value of the filter to zero in one single step such that the desired sparsity degree is reached. The number of zero coefficients is chosen so that the sparse filter has exactly the same amount of nonzero coefficients as some full filter. This way a comparison can be made between the maximum absolute value of the error for a high order sparse filter and for a smaller order full filter. For example, for size 29×29, from the 841 coefficients just 361 are nonzero because the comparison is made with the full filter of size 19×19. After the zero coefficients are fixed, the filter is optimized on the support set by solving (10) again with the selected set S and no weights. The filters are generated using the parameters: maxSteps $=$ 15, $\epsilon = 10^{-5}$, $\epsilon^{\text{stop}} = 10^{-4}$, $\epsilon^{\text{cut}} = 10^{-6}$, $a = 4$ and δ^{imposed} the value obtained by using the method from [14]. For each filter designed using the method from [14] a sweep is done with (10) for several values of μ_1 to identify the one that gives the best results. For our approach a different parameter μ_2 is used. The values used for the μ parameters are given in Table 4. Table 5 and Table 6 describe the results obtained using the diamond and circular

Table 5: 2D sparse diamond shaped lowpass filters simulations results

	nonzero coefficients				δ^{minimax}			
N	IRL1G×3	IRL1G	[14]	OFF	IRL1G×3	IRL1G	[14]	OFF
29	317	323	361	361	0.000921	0.000966	0.000984	0.00210
23	199	205	225	225	0.00366	0.00365	0.00373	0.00553
17	165	169	169	169	0.00536	0.00539	0.00539	0.01076
11	43	43	49	49	0.08075	0.08075	0.08077	0.08733

Table 6: 2D sparse circular shaped lowpass filters simulations results

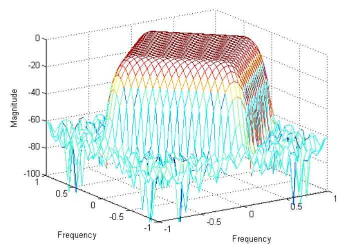
	nonzero coefficients				δ^{minimax}			
N	IRL1G×3	IRL1G	[14]	OFF	IRL1G×3	IRL1G	[14]	OFF
29	347	349	361	361	0.00812	0.00804	0.00812	0.01117
23	221	221	225	225	0.01818	0.01818	0.01827	0.02871
17	165	165	169	169	0.02895	0.02895	0.02942	0.03609
11	49	49	49	49	0.11599	0.11599	0.11892	0.15853

shaped low pass filters and a comparison with the optimal full filters (OFF) of appropriate length and [14]. While IRL1G solves on an average $7 - 8$ LPs, the method proposed in [14] solves 2 optimization problems. But, as it can be seen clearly from Table 5, the error is smaller and the number of nonzero coefficients is smaller in our approach, especially for the filters with larger order. On an average, IRL1G×3 runs for about 30 LPs but it is able to reach more zero coefficients in the process. The amplitude responses are depicted in Figure 1 for the diamond shaped lowpass filters with $N = 29$ generated using both approaches.

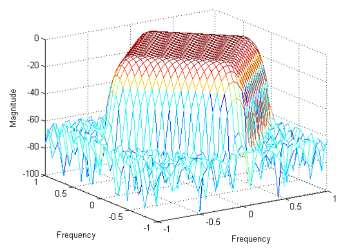
In the case of the $N = 29$ diamond shaped lowpass filter, the running time is about 240 seconds for IRL1G×3 and about 15 seconds for the approach from [14].

5. Conclusions

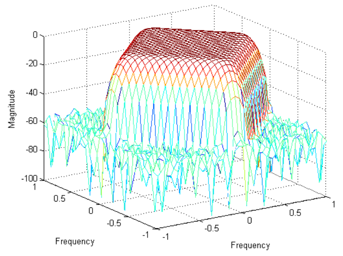
By combining two of the most popular approaches in the field of sparse representations we have developed a new efficient method called IRL1G that can be used to design sparse 1D and 2D FIR filters. The power of IRL1G re-



(a)



(b)



(c)

Figure 1: Amplitude responses of the sparse 2D FIR diamond shaped lowpass filters of size 29×29 generated using the method described in this paper (a) and using the method described in [14] (b). The equivalent full optimized 2D filter of size 19×19 with $\delta^{\text{minimax}} = 0.0021$ and 361 nonzero coefficients in (c).

lies on its two stages: the reweighted l_1 minimization that induces a relatively large number of zero coefficients in just a few iterations and the polishing greedy iterations that run only for a few steps and further eliminate a few extra coefficients. This combination means that the algorithm is fast, even for filters of large size, and experimental results show that it is able to design sparse filters with low errors. A comparison of the obtained results with recent methods for sparse filter design is also supplied showing that IRL1G provides a faster running time while maintaining the performance for the 1D case and provides sparser filters for the 2D case.

6. Acknowledgements

The work has been funded by the Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family and Social Protection through the Financial Agreement POSDRU/88 /1.5/S/61178.

References

[1] M. Bateman and B. Liu, An approach to programmable CTD filters using coefficients 0, +1, and -1, IEEE Trans. Circ. Syst., vol. CAS-27, no. 6 (1980) 451–456.

[2] D. Li, Y. C. Lim, Y. Lian, and J. Song, A polynomial-time algorithm for designing FIR filters with power-of-two coefficients, IEEE Trans. Signal Proc., vol. 50, no. 8 (2002) 1935–1941.

[3] J. Adams and A. Willson, Jr., Some efficient digital prefilter structures, IEEE Trans. Circ. Syst., vol. CAS-31, no. 3 (1984) 260–266.

[4] R. J. Hartnett and G. F. Boudreaux-Bartels, On the use of cyclotomic polynomial prefilters for efficient FIR filter design, IEEE Trans. Signal Proc., vol. 41, no. 5 (1993) 1766-1779.

[5] T. Saramaki, Y. Neuvo, and S. K. Mitra, Design of computationally efficient interpolated FIR filters, IEEE Trans. Circ. Syst., vol. 35, no. 1 (1988) 70–88.

[6] Y. C. Lim and Y. Lian, Frequency-response masking approach for digital filter design: Complexity reduction via masking filter factorization, IEEE Trans. Circ. Syst. II, vol. 41, no. 8 (1994) 518–525.

[7] Y. C. Lim and Y. Lian, The optimum design of one- and two-dimensional FIR filters using the frequency response masking technique. IEEE Trans. Circ. Syst., vol. 40, no. 2 (1993) 88–95.

[8] A. M. Bruckstein, D. L. Donoho and M. Elad, From Sparse Solutions of Systems of Equations to Sparse Modeling of Signals and Images, SIAM Review, vol. 51, no. 1 (2009) 34–81.

[9] S.S. Chen, D.L. Donoho, and M.A. Saunders, Atomic decomposition by basis pursuit, SIAM J. Sci. Comput., vol. 20 (1998) 33–61.

[10] S. Mallat and Z. Zhang, Matching pursuits with time-frequency dictionaries, IEEE Trans. Signal Proc., vol. 41, (1993) 3397–3415.

[11] D. Mattera, F. Palmieri, and S. Haykin, Efficient sparse FIR filter design, IEEE Int. Conf. Acoust., Speech, Signal Process., vol. 2 (2002) 1537-1540.

[12] T. Baran, D. Wei and A. V. Oppenheim, Linear Programming Algorithms for Sparse Filter Design, IEEE Trans. Signal Proc., vol. 58, no. 3 (2010) 1605–1617.

[13] J. L. H. Webb and D. C. Munson Jr., Chebyshev optimization of sparse FIR filters using linear programming with an application to beamforming, IEEE Trans. Signal Proc., vol. 44, no. 8 (1996) 1912-1922.

[14] W. S. Lu and T. Hinamoto, Two-dimensional digital filters with sparse coefficients, Multidim. Syst. Signal Proc., vol. 22, issue 1-3 (2011) 173–189.

[15] E. J. Candes, M. B. Wakin and S. Boyd, Enhancing Sparsity by Reweighted l1 Minimization, J. Fourier Analysis and Applications, vol. 14, no. 5 (2008) 877–905.