FIR Filter Design Based on Successive Approximation of Vectors

Eduardo A. B. da Silva, *Senior Member IEEE* Lisandro Lovisolo, *Member IEEE,*
Alessandro J. S. Dutra, *Member IEEE,* Paulo S. R. Diniz, *Fellow IEEE*

***Abstract*—We present a novel method for the design of finite impulse response (FIR) filters with discrete coefficients that belong in the sum of powers-of-two (POT) space. The importance of this class of filters cannot be overstated, given the ever increasing number of applications for which a specific hardware implementation is needed. Filters which have coefficients that belong to such a class are also referred to as *multiplierless* filters, given that the operations performed by the filter can all be implemented by using appropriately designed shifts of the input data, making them a perfect choice whenever implementation simplicity and processing speed are the ultimate goal. To produce such a design, we employ a vector successive approximation technique successfully used in data compression that has a very low computational complexity, the *Matching Pursuits Generalized BitPlanes* algorithm (MPGBP). We derive optimality conditions for the approximation dictionary. We compare filters obtained with the proposed method with those derived in previous works. Based on this comparative analysis, we show that this new and powerful way of producing the filters' coefficients is also among the simplest available in the literature.**

I. Introduction

It has long been a subject of interest to design FIR filters whose coefficients belong to a constrained class of numbers, in particular the sums of powers-of-two (SOPOT) class [1]–[4], as opposed to using the high-precision floating-point coefficients generated by conventional methods. This approach is particularly useful for hardware implementations [2,5,6], commonly referred to as *multiplier-less* design [1,3,5,7,8], in which all coefficients are generated from a combination of shifts and adds/subtracts, thereby eliminating the hardware cost associated with a general multiplier. In addition, such design strategy together with a generic hardware implementation allows to adapt and program filters on the fly [5].

In the search for such useful representations, several methods have been developed [1,3,5]–[14]. For example, integer programming has been used to produce solutions to minimization problems using discrete constraints in the filter coefficient values for both the weighted minimax and weighted least-squares objective functions [11,12]. In either case, the filter designs obtained are optimal for a given word length, but the design processes are extremely complex in computational terms. More recently, a method using linear programming to optimize the coefficients directly in the sub-expression space was proposed [9] that yields a considerable reduction in the number of required sums to attain a given specification. However, SOPOT filters impose additional restrictions on the design of quantized coefficients filters [15]–[18], an important aspect is how to allocate the available limited hardware non-uniformly among coefficients in order to obtain the desired performance [14]. Due to the complex nature of the SOPOT-Filter coefficients optimization other common optimization approaches have also been tried, as for example Genetic Algorithms [8,13]. A common drawback of these approaches rests on the design time [14] that impairs the practical use of SOPOT programmable filters. A different optimization approach is used in [7], in which an infinite-precision prototype filter is designed to exceed the prescribed specifications, thereby producing a margin of error for the coefficient quantization. A non-linear optimization is then performed to determine the final filter representation. In [19] following some relaxation on the filter specifications an heuristic for designing filters is presented that obtain SOPOT filters. In particular [20] emphasizes the problem of obtaining long filters using the available methods, proposing an approach based on the design of extrapolated impulse responses for FIR filters.

In this paper, we propose a design method also based on the approximation of an infinite precision prototype filter, designed to match or exceed the prescribed specifications. We employ a technique, initially developed for data compression applications, that decomposes vectors in generalized bitplanes using successive approximations. This technique allows us to define the representation dictionary in such a way that the design computational complexity is kept to a minimum. A striking feature of the new proposed design method is its simple implementation such that no sophisticate optimization or exhaustive search programs are required.

The paper is organized as follows: Section II presents a brief review of successive approximation vector representation techniques, including the *Matching Pursuits Generalized Bit-Plane* (MPGBP) algorithm [21], which will serve as the basis for our proposed approximation method. It also presents some novel theoretical results specific to the variation of MPGPB used in this paper. The proposed method for the successive approximation design of SOPOT FIR filters is then described in Section III. Section IV presents a theoretical analysis of the performance of the proposed MPGBP variation in the context of SOPOT FIR filter design, used to establish some design constraints. Section V describes in detail the design algorithm along with an analysis of its complexity, a relevant discussion when greedy algorithms as the one employed here are used for practical purposes. Examples and performance comparisons against other existing methods are discussed in Section VII. Our conclusions and future directions are detailed in Section VIII. Appendix A contains the convergence proof for the algorithm presented in Section III.

II. Successive Approximation of Vectors

Several methods for the approximation of vectors have been successfully proposed for use in image and video coding fields [22]. The existence of particular constraining factors in those applications, such as the ever present trade-off on permissible encoding rate *vs.* reproduction quality [23], have led us to conjecture that the use of those algorithms for approximating the coefficients of FIR filters, where the main constraint lies on whether or not the filter frequency response satisfies the specifications [24], might enjoy similar success.

We shall start this section with some brief definitions to establish the notation that will be used in the remaining of the article. We then proceed to review the Matching Pursuits algorithm [25] and, in Sec. II-C, we show how it can be modified into a more structured decomposition method with the addition of the generalized bit-planes concept.

A. Notation

For a vector $\boldsymbol{v}$, $v(i)$ represents its i-th component. If $\boldsymbol{u}$ and $\boldsymbol{v}$ are N-dimensional complex-valued vectors, the inner product of $\boldsymbol{u}$ and $\boldsymbol{v}$ will be denoted $\langle \boldsymbol{u}, \boldsymbol{v} \rangle$ and has the usual definition

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \boldsymbol{u}^{\mathrm{T}} \boldsymbol{v}^{*} = \sum_{i=0}^{N-1} u(i) v^{*}(i), \tag{1}$$

where $v^{*}(i)$ is the complex conjugate of $v(i) \in \mathbb{C}$ ([26]). The norm of $\boldsymbol{u}$ is then defined as

$$\|\boldsymbol{u}\| = (\langle \boldsymbol{u}, \boldsymbol{u} \rangle)^{1/2} = (\boldsymbol{u}^{\mathrm{T}} \boldsymbol{u}^{*})^{1/2}. \tag{2}$$

We describe an N-th order FIR filter both in terms of its impulse response

$$H(z) = h(0) + h(1) z^{-1} + \ldots + h(n) z^{-N}, \tag{3}$$

and its vector notation

$$\boldsymbol{h} = [h(0)\, h(1)\, \ldots\, h(N)]^{\mathrm{T}}, \tag{4}$$

so that when the filter coefficients are real-valued scalars, $h(i) \in \mathbb{R}$, $\boldsymbol{h}$ is an $(N+1)$-dimensional column vector, i.e. $\boldsymbol{h} \in \mathbb{R}^{N+1}$.

B. The Matching Pursuits Algorithm

The Matching Pursuits (MP) algorithm presents a suboptimal solution to the NP-hard problem of finding the best expansion for a function given a redundant, overcomplete, dictionary of waveforms or codebook [25,27]–[29]. Instead of searching over all the possible combinations for the lowest possible distortion, the MP method builds an expansion based on an iterative search. In each iteration, one searches for the dictionary word (element or atom) that is closest to the current representation residue, in terms of an appropriately defined distortion measure – in the first iteration the residue is the function itself. In this article, we consider a simpler, particular instance of the expansion problem, that of approximating an N-dimensional vector given a large, redundant dictionary, that expands $\mathbb{R}^N$. Formally, if a vector $\boldsymbol{x} \in \mathbb{R}^N$ is to be represented using a dictionary $\mathcal{D} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_Q\}$, $\boldsymbol{c}_j \in \mathbb{R}^N$, $\|\boldsymbol{c}_j\| = 1$, the MP method proceeds as described in Algorithm 1.

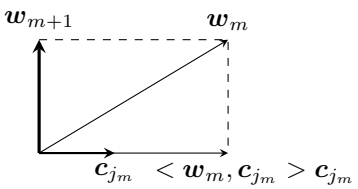

Fig. 1. MP m-th decomposition step

Algorithm 1: The Matching Pursuits (MP) Algorithm [25]

1. Start with $m = 1, \boldsymbol{r}_1 = \boldsymbol{x}$
2. Repeat until a stop criterion is met:
   (a) Find the closest codeword, i.e., find $j_m \in \{1, \ldots, Q\}$ such that
   $$\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle = \max_{1 \le j \le Q} \{ |\langle \boldsymbol{r}_m, \boldsymbol{c}_j \rangle| \}$$
   (b) Choose
   $$k_m = \langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle$$
   (c) Let $\boldsymbol{r}_{m+1} = \boldsymbol{r}_m - k_m \boldsymbol{c}_{j_m}$
   (d) Increment m
3. Stop.

■

At each step m, including the initial one, the approximation residue will be denoted $\boldsymbol{r}_m$ and the expansion rule produces

$$\boldsymbol{r}_m = \langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle \boldsymbol{c}_{j_m} + \boldsymbol{r}_{m+1}. \tag{5}$$

Throughout this work one employs a subscripted symbol as in μ_m to denote the value of μ selected in iteration m. Although subscripted symbols are also used for indexing elements in a set, as in the dictionary definition, their appropriate interpretation will be clear from the context.

The expansion rule in eq. (5) provides the representation

$$\boldsymbol{x} = \sum_{m=1}^{M} \langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle \boldsymbol{c}_{j_m} + \boldsymbol{r}_{M+1}. \tag{6}$$

In eq. (5), $\boldsymbol{r}_{m+1}$ is orthogonal to $\boldsymbol{c}_{j_m}$, as depicted in Figure 1. Therefore, we can write

$$\|\boldsymbol{r}_m\|^2 = |\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle|^2 + \|\boldsymbol{r}_{m+1}\|^2. \tag{7}$$

Eq. (7) can be further extended to include all terms of the decomposition up to the M-th step, yielding $\|\boldsymbol{x}\|^2 = \sum_{m=1}^{M} |\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle|^2 + \|\boldsymbol{r}_{M+1}\|^2$. For more details, the reader is referred to [25]. We present a modified version of the MP algorithm which is used as the main engine in the impulse response approximation method presented in this work.

C. Matching Pursuits with Generalized Bit-planes

Each step of the approximation process in the MP algorithm produces two pieces of information, namely the index of the closest dictionary (codebook) codeword and the length of the projection of the residue onto that codeword. That is, using K decomposition steps the MP algorithm provides the M-term signal representation:

$$\boldsymbol{x} \approx \boldsymbol{x}^{(M)} = \sum_{m=1}^{M} \langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle \boldsymbol{c}_{j_m}, . \tag{8}$$

where $\boldsymbol{c}_{j_m} \in \mathcal{D} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_Q\}$ and j_m indexes the dictionary codeword selected at the k-th decomposition step.

If one expands $\mathcal{D}$ to $\overline{\mathcal{D}} = \{\pm\boldsymbol{c}_1, \pm\boldsymbol{c}_2, \ldots, \pm\boldsymbol{c}_Q\}$, one can guarantee that $\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle > 0$ for $\boldsymbol{c}_m \in \overline{\mathcal{D}}$. There are some works that deal with MP approximation ratio [29,30] and convergence [31].

However, in practical applications, the length information, given by $\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle$, needs to be quantized, and the above cited approximation ratio and convergence analyses need to be extended. In [21] a version of the MP algorithm has been proposed such that this length is quantized as an integer power k_m of a parameter α, $\alpha < 1$, that is,

$$\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle \approx \alpha^{k_m}, \quad k_m \in \mathbb{Z}. \tag{9}$$

The value of k_m above is chosen such that $\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle$ is closer to α^{k_m} than it is to α^{k_m-1} or α^{k_m+1}. Applying the nearest neighbor optimal quantization rule [23], this is equivalent to

$$\frac{\alpha^{k_m+1} + \alpha^{k_m}}{2} \leq \langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle < \frac{\alpha^{k_m} + \alpha^{k_m-1}}{2} \tag{10}$$

$$\alpha^{k_m} \leq \frac{2}{1+\alpha} \langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle < \alpha^{k_m-1}$$

$$k_m - 1 < \log_\alpha \frac{2\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle}{1+\alpha} \leq k_m \text{ (as } \alpha < 1) \tag{11}$$

$$\text{and thus } k_m = \left\lceil \log_\alpha \left(\frac{2\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle}{1+\alpha} \right) \right\rceil \tag{12}$$

where $\lceil y \rceil$ is the smallest integer larger than or equal to y. The complete algorithm is described in Algorithm 2 below. Note that, in order to avoid error accumulation, the residue in step m is computed using the quantized value of the projection (step 2c in Algorithm 2). We refer to this algorithm as the *Matching Pursuits with Generalized Bit-Planes* (MPGBP) algorithm. This name comes from the fact that we refer to the set of $\boldsymbol{c}_{j_m}$ such that $k_m = k$ as the *g*eneralized bitplane k.

Algorithm 2: The MPGBP Algorithm

1. Given $\alpha < 1$, start with $m = 1$, $\boldsymbol{r}_1 = \boldsymbol{x}$
2. Repeat until a stop criterion is met:
   (a) Choose $j_m \in \{1, \ldots, Q\}$ such that
   $$\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle = \max_{1 \leq j \leq Q} \{\langle \boldsymbol{r}_m, \boldsymbol{c}_j \rangle\}$$
   (b) Choose
   $$k_m = \left\lceil \log_\alpha \left(\frac{2\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle}{1+\alpha} \right) \right\rceil$$
   (c) Replace
   $$\boldsymbol{r}_{m+1} \leftarrow \boldsymbol{r}_m - \alpha^{k_m} \boldsymbol{c}_{j_m}$$
   (d) Increment m.
3. Stop.

■

The MPGBP algorithm produces, after M steps, an approximation $\boldsymbol{x}^{(M)}$ for $\boldsymbol{x}$ given by

$$\boldsymbol{x}^{(M)} = \sum_{m=1}^{M} \alpha^{k_m} \boldsymbol{c}_{j_m}, \tag{13}$$

where $\boldsymbol{c}_{j_m} \in \mathcal{D}$. As pointed out in the commentary after eq. (8), since $\alpha^{k_m} > 0$, $\mathcal{D}$ must be such that if $\boldsymbol{c}_j \in \mathcal{D}$ then necessarily $-\boldsymbol{c}_j \in \mathcal{D}$. Also, each $\boldsymbol{c}_j$ has unit-norm. The convergence properties of the MPGBP algorithm are given by Theorem 1.

Theorem 1 (MPGBP Convergence): Given a dictionary $\mathcal{D}$ such that if $\boldsymbol{c}_j \in \mathcal{D}$ then necessarily $-\boldsymbol{c}_j \in \mathcal{D}$ and also $\|\boldsymbol{c}_j\| = 1$, then for $\alpha < 1$ and any input vector $\boldsymbol{x}$ the MPGBP Algorithm (Algorithm 2) converges, and the error incurred in the approximation $\boldsymbol{x}^{(M)}$ is bounded by

$$\left\| \boldsymbol{x} - \boldsymbol{x}^{(M)} \right\|^2 \leq \beta^M \|\boldsymbol{x}\|, \tag{14}$$

$$\text{where } \beta = \sqrt{1 - \frac{4\alpha}{(1+\alpha)^2} \cos^2(\Theta(\mathcal{D}))}, \tag{15}$$

$$\text{and } \Theta(\mathcal{D}) = \cos^{-1} \left\{ \min_{\boldsymbol{x} \in \mathbb{R}^N} \left\{ \max_{\boldsymbol{c} \in \mathcal{D}} \left(\frac{\langle \boldsymbol{x}, \boldsymbol{c} \rangle}{\|\boldsymbol{x}\| \, \|\boldsymbol{c}\|} \right) \right\} \right\}. \tag{16}$$

□

$\Theta(\mathcal{D})$ is the largest angle between any vector $\boldsymbol{x}$ from the considered signal space and its closest codeword $\boldsymbol{c}_i$ [21,32,33]. The proof of Theorem 1 is presented in Appendix A. ■

A suboptimal version of the MPGBP has been presented without proof in [21] for video coding. In that setting, the value of k_m was given by

$$k_m = \lceil \log_\alpha (\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle) \rceil. \tag{17}$$

III. FIR Filter Representation using SOPOTs

We now turn to the presentation of the complete filter design method, which includes the prototype filter design, its approximation and the representation of its coefficients as sums of powers-of-two.

A. FIR Filter Approximation

Given a set of frequency response specifications $\{\omega_p, \omega_s, \delta_p, \delta_s\}$, corresponding to the passband and stop-band edges and their allowed undulations respectively, we start by designing an order $N-1$ prototype filter $h(n)$, which will be represented as $\boldsymbol{h} = \begin{bmatrix} h(0) & h(1) & \cdots & h(N-1) \end{bmatrix}^T$. Our proposal is to approximate the coefficients of $\boldsymbol{h}$ using sums of powers-of-two through the MPGPB algorithm. If we use $\alpha = \frac{1}{2}$, then eq. (13) becomes, for coordinate n of $\boldsymbol{x}^{(M)}$,

$$x^{(M)}(n) = \sum_{m=1}^{M} 2^{-k_m} c_{j_m}(n), \quad \text{for } n \in [0, N-1]. \tag{18}$$

We can see from the above equation that if one restricts the coordinates $c_{j_m}(n)$ of the codewords $\boldsymbol{c}_{j_m}$ to be either -1, 1 or 0, eq. (18) gives a binary representation of the coordinates of $\boldsymbol{x}^{(M)}$. Therefore, in order to approximate the coefficients of $\boldsymbol{h}$ using the MPGBP algorithm, we should define the approximation dictionary

$$\mathcal{D} = \{\pm\boldsymbol{g}_1, \pm\boldsymbol{g}_2, \ldots, \pm\boldsymbol{g}_Q\}, \tag{19}$$

in which the codewords $\boldsymbol{g}_i \in \mathbb{R}^N$, and each codeword has P components equal to ± 1 and $(N-P)$ components equal to 0, i.e., they are permutations of the form

$$\boldsymbol{g}_i = \begin{bmatrix} \pm 1^P & 0^{(N-P)} \end{bmatrix}^T := [\underbrace{\pm 1 \ldots \pm 1}_{P \text{ times}} \underbrace{0 \ 0 \ \ldots \ 0 \ 0}_{(N-P) \text{ times}}]^T. \tag{20}$$

An example of such a codeword, for $N = 7$ and $P = 3$ is the vector $\begin{bmatrix} 1 & 0 & 1 & -1 & 0 & 0 & 0 \end{bmatrix}^T$.

At first, one could argue that such a dictionary cannot be used with the MPGPB algorithm because its codewords have

norm $\|\boldsymbol{c}_{j_m}\| = \sqrt{P} \neq 1$. However, if in eq. (12) and step 2b of Algorithm 2, one replaces $\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle$ by $\frac{1}{\|\boldsymbol{c}_{j_m}\|^2}\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle = \frac{1}{P}\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle$, then Algorithm 2 will still generate an approximation of $\boldsymbol{x}^{(M)}$ as in eq. (13) as this replacement is equivalent to using unit-norm atoms. However, this replacement can be dropped in implementation. This is so, because all codewords defined according to eq. (20) for given P and N, have the same norm, not affecting the largest correlated codeword criterion of MP. Therefore, each of the coordinates $x^{(M)}(n)$ of $\boldsymbol{x}^{(M)}$ ends up being approximated as in eq. (18).

The vector $\boldsymbol{x}$ to be approximated using Algorithm 2 is initialized with the filter prototype $\boldsymbol{h}$. Since $\alpha = \frac{1}{2}$ and the norm of the code-vectors is P, the computation of k_m in step 2.b of Algorithm 2 becomes

$$k_m = \left\lceil -\log_2 \left(\frac{4\langle \boldsymbol{r}_m, \boldsymbol{g}_{j_m} \rangle}{3P} \right) \right\rceil . \tag{21}$$

Due to the particular structure of the codewords (eq. (20)), the determination of the codeword $\boldsymbol{g}_{j_m}$ closest to the residual $\boldsymbol{r}_m$ (step 2a of Algorithm 2) does not require an exhaustive search as in ordinary vector quantization algorithms [23]. It can be accomplished in an efficient way, as described next.

Since the coordinates of the code-vectors are either -1, 1 or 0, the inner product $\langle \boldsymbol{r}_m, \boldsymbol{g}_{j_m} \rangle$ is equivalent to adding the coordinates of $\boldsymbol{r}_m$ corresponding to the coordinates of $\boldsymbol{g}_{j_m}$ equal to 1 and subtracting the coordinates of $\boldsymbol{r}_m$ corresponding to the coordinates of $\boldsymbol{g}_{j_m}$ equal to -1. If one wants the largest inner product, two conditions must be satisfied:

(i) The corresponding coordinates of $\boldsymbol{r}_m$ and $\boldsymbol{g}_{j_m}$ should have the same sign.
(ii) The coordinates of largest magnitude should correspond to either 1 or -1.

With these conditions in mind, a fast algorithm for finding the closest vector $\boldsymbol{g}_{j_m}$ to a vector $\boldsymbol{r}_m$ can be as described in Algorithm 3 below.

Algorithm 3: Fast Algorithm for finding the closest codeword to $\boldsymbol{r}$ in $\mathcal{D}$ with codewords $\boldsymbol{c}_j$ having P coordinates ± 1 and $N - P$ coordinates 0 (permutations of eq. (20)):

1. Sort the absolute values of the coordinates of the current representation of $\boldsymbol{r}$ in decreasing order of magnitude and store the indexes of the P largest ones.
2. The approximation codeword is obtained by setting those P coordinates whose indexes were stored in step 1 to $+1$ if the coordinate is positive and to -1 if the coordinate is negative. The remaining coordinates are set to zero.

■

Example 1: Vector approximation using the MPGBP algorithm. – Suppose the vector $\boldsymbol{h} = [0.25, 0.5, 0.5, 0.75]$ is to be approximated using the proposed method using a codebook $\mathcal{D}$ with $P = 2$. In this case, $N = 4$, which implies that the codebook is comprised of codewords which are permutations of $[\pm 1, \pm 1, 0, 0]$. Initially, we set $\boldsymbol{r}_1 = \boldsymbol{h}$. The steps taken by the algorithm are presented in Table I. From Table I, $\boldsymbol{h}$ is approximated as

$$\boldsymbol{h}^{(3)} = 2^{-1}\begin{bmatrix}0\\1\\0\\1\end{bmatrix} + 2^{-1}\begin{bmatrix}1\\0\\1\\0\end{bmatrix} + 2^{-2}\begin{bmatrix}-1\\0\\0\\1\end{bmatrix}. \tag{22}$$

We see that, in this particular example, the method produces an error-free representation of the input vector in just 3 steps, using at most one adder per filter coefficient (e.g., $h(3) = 0.75 = 2^{-1} + 2^{-2}$). □

TABLE I
EXAMPLE OF USE OF THE MPGBP ALGORITHM

m	$\boldsymbol{r}_m$	$\boldsymbol{g}_{j_m}$	k_m	$\boldsymbol{h}^{(m)}$
1	$[0.25\ 0.5\ 0.5\ 0.75]^{\mathrm{T}}$	$[0\ 1\ 0\ 1]^{\mathrm{T}}$	-1	$[0\ 0.5\ 0\ 0.5]^{\mathrm{T}}$
2	$[0.25\ 0\ 0.5\ 0.25]^{\mathrm{T}}$	$[1\ 0\ 1\ 0]^{\mathrm{T}}$	-1	$[0.5\ 0.5, 0.5\ 0.5]^{\mathrm{T}}$
3	$[-0.25\ 0\ 0\ 0.25]^{\mathrm{T}}$	$[-1\ 0\ 0\ 1]^{\mathrm{T}}$	-2	$[0.25\ 0.5\ 0.5\ 0.75]^{\mathrm{T}}$
4	$[0, 0, 0, 0]^{\mathrm{T}}$			

B. Common-terms Reduction Step

Since $2^{-k} + 2^{-k} = 2^{-k+1}$, a representation using sums of powers-of-two is in general not unique. In filter design, we aim at the representation corresponding to the smallest complexity, and therefore, with the smallest number of terms. However, the MPGBP algorithm may generate a representation for filter coefficients that does not correspond to the smallest number of powers-of-two, that is, it contains terms that may be combined into a different power-of-two, as illustrated in Table II. Therefore, in order to generate a representation with the minimum number of terms, we must carry out some post-processing to the output of the MPGBP algorithm. We refer to it as the *Common-terms Reduction Step*.

This post-processing operation starts from the representation of the coefficient $\boldsymbol{h}(n)$ after the Mth iteration, $h^{(M)}(n)$, and finds the minimal representation according to Algorithm 4. This algorithm finds a minimal representation for numbers that can be expressed exactly as a finite sum of powers-of-two, the ones stored in k_i in the algorithm. This is the case for $h^{(M)}(n)$, since it is output by the MPGBP algorithm.

TABLE II
COMMON-TERMS REDUCTION EXAMPLES

$h(i)$	Initial Representation	Minimal Representation	Number of Adders (Initial → Minimal)
0.75	$\frac{1}{2} + \frac{1}{8} + \frac{1}{8}$	$\frac{1}{2} + \frac{1}{4}$	$2 \rightarrow 1$
0.5	$\frac{1}{8} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8}$	$\frac{1}{2}$	$3 \rightarrow 0$
0.5	$\frac{1}{4} + \frac{1}{4}$	$\frac{1}{2}$	$1 \rightarrow 0$
0.25	$\frac{1}{8} + \frac{1}{8}$	$\frac{1}{4}$	$1 \rightarrow 0$

Algorithm 4: Common-Terms Reduction Algorithm
For each $h^{(M)}(n)$, $0 \leq n \leq N-1$, in $\boldsymbol{h}^{(M)}$:
1. Set $x = h^{(M)}(n)$, $i = 0$
2. Repeat until $x = 0$
   (a) i = i+1
   (b) Evaluate $k_i = \lfloor \log_2 x \rfloor$
   (c) Let $x \leftarrow x - 2^{k_i}$

■

The search for the best approximation for a given filter specification may include testing for prototypes of different orders and dictionaries having different values of P, the number of non-zero codeword components. The implementation chosen may be the one that yields the lowest implementation complexity in terms of quantity of SOPOTs used by the filter, while satisfying the given specifications.

The design process may be stopped at any time. However, usual stop criteria include the approximation error and/or the number of additions / subtractions used in the approximation. It is also worth noticing that the number of POTs used in the approximation of distinct components is not fixed, a constraint usually found in other optimization methods. One should note that sometimes a higher order filter may end up with a lower complexity representation considering the number of adders resulting from the design process. In the examples given in this paper, the filter prototypes are designed using the Parks-McClellan algorithm [24]. In addition, a global output multiplier is determined to ensure that the average passband gain is kept at 0 dB to allow for straightforward comparisons related to the obtained filter selectivity.

IV. Performance Bounds and Design Issues

When approximating an impulse response $\boldsymbol{h}$ using eq. (18) through the MPGBP algorithm, the main objective is to minimize the number of sum of powers-of-two. This can be accomplished by minimizing the number of steps M necessary to achieve a given approximation error ϵ, that is,

$$\frac{1}{2\pi}\int_{-\pi}^{\pi}\left|H(e^{\jmath\omega}) - H^{(M)}(e^{\jmath\omega})\right|^2 d\omega = \left\|\boldsymbol{h} - \boldsymbol{h}^{(M)}\right\|^2 \leq \epsilon. \tag{23}$$

From Theorem 1, eqs. (14) and (15), the approximation error after M steps is bounded by $\beta^M \|\boldsymbol{h}\|$, where, for $\alpha = \frac{1}{2}$,

$$\beta = \sqrt{1 - \frac{8}{9}\cos^2\left(\Theta(\mathcal{D})\right)}, \tag{24}$$

Therefore, in order to minimize M one must have a codebook with the smallest possible value of $\Theta(\mathcal{D})$. In this section we find $\Theta(\mathcal{D})$ for the codebooks having codewords as in eq. (20) and discuss design issues.

A. Computation of $\Theta(\mathcal{D})$

We saw in Section III-A that the MPGBP algorithm will use a codebook whose codewords $\boldsymbol{g}$ are permutations of the form in eq. (20), i.e.

$$\left[\pm 1^P, 0^{N-P}\right]. \tag{25}$$

In this section we determine the minimum attainable value for $\Theta(\mathcal{D})$ given P and N – the number of non-zero terms P and dimension N. Since the codebook contains all the permutations of the form described by eq. (20), we can suppose, without any loss of generality, that the vector $\boldsymbol{x} \in \mathbb{R}^N$ making an angle of $\Theta(\mathcal{D})$ with the codeword $\boldsymbol{c}_j$ closest to it can be represented as

$$\boldsymbol{x} = [x(1), x(2), \ldots, x(P), a, a, \ldots, a] \tag{26}$$

$$\text{where } x(1) \geq x(2) \geq \ldots \geq x(P) \geq a \geq 0, \tag{27}$$

that is, the vector components are sorted in decreasing order and they are all non-negative. This is so because (note that for the sake of the $\Theta(\mathcal{D})$ computation, we can restrict $\boldsymbol{x}$ to be on a hyper-sphere, that is, to have a predetermined norm [33]):

(i) For any permutation of $\boldsymbol{x}$ in eq. (26) the closest codeword can be obtained by applying the same permutation to $\boldsymbol{c}_j$, with the value of inner product, and therefore of $\Theta(\mathcal{D})$, remaining the same.
(ii) If we change the sign of some of the components of $\boldsymbol{x}$, then the closest codeword can be obtained from $\boldsymbol{c}_j$ by changing their $+1$ coordinates to -1 whenever the corresponding $\boldsymbol{x}$ coordinates are negative.
(iii) The values of the coordinates of $\boldsymbol{x}$ corresponding to the coordinates of $\boldsymbol{c}_j$ that are zero do not affect their inner product. In addition, the largest inner product occurs if the coordinates of $\boldsymbol{x}$ corresponding to the P non-zero coordinates of $\boldsymbol{c}_j$ are the largest ones.
(iv) If $\boldsymbol{x}$ was such that $\boldsymbol{x} = [x(1),\ x(2),\ \ldots,\ x(P),\ y(1),\ y(2),\ \ldots,\ y(N-P)]$ where $x(j) \geq y(i) \geq 0, \forall(i,j)$, such that $\sum_{i=1}^{N-P} y^2(i) = (N-P)a^2$, then $\langle \boldsymbol{x}, \boldsymbol{c}_j \rangle$ and $\|\boldsymbol{x}\|$ (and thus, the angle) would remain unaltered.

To approximate the vector $\boldsymbol{x}$ as defined in eq. (26) using codewords from a codebook $\mathcal{D}$, with elements $\boldsymbol{g}$ as in eq. (20), that have a fixed norm $\sqrt{P}$, we consider that $\boldsymbol{x}$ has unit norm without any loss of generality and, therefore, we may write

$$\sum_{i=1}^{P} x^2(i) + (N-P)a^2 = 1. \tag{28}$$

This assumption leads to the following expression for $\Theta(\mathcal{D})$ (assuming that both $\boldsymbol{g}$ and $-\boldsymbol{g}$ belong to $\mathcal{D}$):

$$\begin{aligned}\Theta(\mathcal{D}) &= \max_{\boldsymbol{x},\boldsymbol{g}\in\mathcal{D}} \boldsymbol{x}\angle\boldsymbol{g} = \arccos\left(\min_{\boldsymbol{x},\boldsymbol{g}\in\mathcal{D}} \cos(\boldsymbol{x}\angle\boldsymbol{g})\right) \\ &= \arccos\left(\min_{\boldsymbol{x},\boldsymbol{g}\in\mathcal{D}} \frac{\langle \boldsymbol{x}, \boldsymbol{g}\rangle}{\|\boldsymbol{x}\|\|\boldsymbol{g}\|}\right) = \arccos\left(\min_{\boldsymbol{x},\boldsymbol{g}\in\mathcal{D}} \frac{\sum_{i=1}^{N} g(i)x(i)}{\|\boldsymbol{x}\|\|\boldsymbol{g}\|}\right) \\ &= \arccos\left(\min_{\boldsymbol{x}} \frac{\sum_{i=1}^{P} x(i)}{\sqrt{P}}\right),\end{aligned} \tag{29}$$

since $\boldsymbol{x}$ is defined as in eq. (26) and $\boldsymbol{g}$ (the codeword in the codebook) as in eq. (20).

To verify the behavior of θ, i.e., to look for points in which $\cos(\theta)$ attains its maximum or minimum value, subject to the constraint imposed by eq. (28), we define the Lagrangian functional $f(\cdot)$ as:

$$f(x_1, \ldots, x_P, a, \lambda) = \frac{1}{\sqrt{P}}\sum_{i=1}^{P} x(i) + \lambda\left(\sum_{i=1}^{P} x^2(i) + (N-P)a^2 - 1\right) \tag{30}$$

and equate its partial derivatives with relation to both a and

$x(i)$ to zero. By doing that, we obtain:

$$\frac{\partial f(\cdot)}{\partial x(i)} = 0 \;\Rightarrow \frac{1}{\sqrt{P}} + 2\lambda x(i) = 0 \;\Rightarrow x(i) = -\frac{1}{2\sqrt{P}\lambda}, \tag{31}$$

$$\frac{\partial f(\cdot)}{\partial a} = 0 \;\Rightarrow 2a(N-P) = 0 \;\Rightarrow a \begin{cases} = 0 & \text{if } N \neq P, \\ \text{is irrelevant} & \text{if } N = P. \end{cases} \tag{32}$$

Combining the results of eqs. (31) and (32), we have that

$$\sum_{i=1}^{P} x(i)^2 = 1 \;\Rightarrow\; \frac{1}{4P\lambda^2} P = 1 \;\Rightarrow\; \lambda = \pm\frac{1}{2}. \tag{33}$$

Since $x(i) \geq 0$, $x(i) = \frac{1}{\sqrt{P}}$, we obtain $\cos(\theta) = 1$, which is clearly a maximum.

However, according to the Kuhn-Tucker conditions [34], in order to guarantee the minimization of the objective function we have also to look for solutions lying on the boundary of the convex region delimited by $x(i) = a$ for any i, or $x(i) = 0$ for any i, or $a = 0$. Considering eq. (27), this region is delimited by the cases

$$x(P) = a,\; x(j) \neq a,\; 1 \leq j < P, \tag{34}$$

$$x(P) = x(P-1) = a,\; x(j) \neq a,\; 1 \leq j < P-1, \tag{35}$$

$$\vdots \qquad\qquad \vdots$$

$$x(P) = x(P-1) = \cdots = x(2) = a,\; x(j) \neq a,\; 1 \leq j < 2, \tag{36}$$

$$x(P) = x(P-1) = \cdots = x(1) = a, \tag{37}$$

where $a \geq 0$. The above conditions can be restated as $x(P) = x(P-1) = \ldots = x(P-J+1) = a$, and $x(j) > a,\; 1 \leq j \leq P-J$ with $a \geq 0$ and $1 \leq P \leq N$. Note that $P = J$ implies that $x(j) = a,\; \forall j$. Therefore, the N-dimensional vector $\boldsymbol{x}$ may then be written as

$$\boldsymbol{x} = \begin{cases} [\; x(1) \;\cdots\; x(P-J) \quad \overbrace{a \;\cdots\; a}^{(N-P+J \text{ times})} \quad]^{\mathrm{T}}, & \text{for } P > J, \quad (38) \\ [\; \overbrace{a \;\cdots\; a}^{(N \text{ times})} \;]^{\mathrm{T}}, & \text{for } P = J. \quad (39) \end{cases}$$

Since we have assumed that $\|\boldsymbol{x}\| = 1$, the above equations imply

$$\|\boldsymbol{x}\|^2 = \begin{cases} \sum_{i=1}^{P-J} x^2(i) + (N-P+J)a^2 = 1, & \text{for } P > J, \quad (40) \\ Na^2 = 1, \Rightarrow a = \frac{1}{\sqrt{N}}, & \text{for } P = J, \quad (41) \end{cases}$$

$$\text{and therefore } \cos(\theta) = \begin{cases} \frac{\sum_{i=1}^{P-J} x(i) + Ja}{\sqrt{P}}, & \text{for } P > J, \quad (42) \\ \frac{Pa}{\sqrt{P}} = \sqrt{\frac{P}{N}}, & \text{for } P = J. \quad (43) \end{cases}$$

From eqs. (39) and (41), the above value occurs for

$$\boldsymbol{x} = \left[\tfrac{1}{\sqrt{N}} \;\; \tfrac{1}{\sqrt{N}} \;\; \cdots \;\; \tfrac{1}{\sqrt{N}}\right]^{\mathrm{T}}. \tag{44}$$

Once again, to find the minimum value of $\cos(\theta)$ for $P > J$, $a > 0$ and $x(j) > a$, $1 \leq j \leq P-J$, the Lagrangian functional becomes $f(\cdot)$ given by

$$f(x(1), \ldots, x(P-J), a, \lambda) = \frac{1}{\sqrt{P}} \sum_{i=1}^{P-J} x(i) + \frac{Ja}{\sqrt{P}} + \tag{45}$$
$$+ \lambda \left(\sum_{i=1}^{P-J} x(i)^2 + (N-P+J)a^2 - 1 \right),$$

and taking derivatives with respect to a and $x(i)$ we obtain:

$$\frac{\partial f(\cdot)}{\partial x(i)} = 0 \;\Rightarrow\; \frac{1}{\sqrt{P}} + 2\lambda x(i) = 0$$
$$\Rightarrow\; x(i) = -\frac{1}{2\sqrt{P}\lambda}, \tag{46}$$

$$\text{and } \frac{\partial f(\cdot)}{\partial a} = 0 \Rightarrow \frac{J}{\sqrt{P}} + 2\lambda(N-P+J)a = 0$$
$$\Rightarrow a = \frac{-J}{2\sqrt{P}\lambda(N-P+J)}. \tag{47}$$

From eqs. (46) and (47), the unit-norm condition $\|\boldsymbol{x}\|^2 = 1$ (eq. (40)) leads to

$$\frac{1}{4P\lambda^2}(P-J) + \frac{(N-P+J)J^2}{4P(N-P+J)^2\lambda^2} = 1,$$
$$\frac{1}{4P\lambda^2}\left[P - J + \frac{J^2}{N-P+J}\right] = 1,$$
$$\text{and hence } \frac{1}{2\sqrt{P}\lambda} = \pm\frac{1}{\sqrt{P-J+\frac{J^2}{N-P+J}}}. \tag{48}$$

From the above equation we obtain the following values for the components a and $x(i)$:

$$a = \frac{\frac{J}{N-P+J}}{\sqrt{P-J+\frac{J^2}{N-P+J}}}, \tag{49}$$

$$\text{and } x(i) = \frac{1}{\sqrt{P-J+\frac{J^2}{N-P+J}}}, \tag{50}$$

for $i = 1, \ldots, P-J$. Applying these at eq. (42), one obtains

$$\cos(\theta) = \frac{1}{\sqrt{P}}\left[\frac{P-J}{\sqrt{P-J+\frac{J^2}{N-P+J}}} + \frac{\frac{J^2}{N-P+J}}{\sqrt{P-J+\frac{J^2}{N-P+J}}}\right]$$
$$= \frac{\sqrt{P-J+\frac{J^2}{N-P+J}}}{\sqrt{P}}. \tag{51}$$

Its derivative with respect to J is:

$$\frac{\partial \cos(\theta)}{\partial J} = \frac{-1 + \frac{(N-P+J)2J - J^2}{(N-P+J)^2}}{\sqrt{2P}\sqrt{P-J+\frac{J^2}{N-P+J}}} = \frac{-\sqrt{2P}^{-1}\frac{(N-P)^2}{(N-P+J)^2}}{\sqrt{P-J+\frac{J^2}{N-P+J}}} \tag{52}$$

The above equation implies that the derivative in eq. (52) is zero for all J if $N = P$, being negative for all values of J if $N \neq P$. If $N = P$, from eq. (51), $\cos(\theta)$ is

$$\cos(\theta) = \frac{\sqrt{P-J+\frac{J^2}{J}}}{\sqrt{P}} = 1, \tag{53}$$

which is a maximum value for $\cos(\theta)$.

For $N \neq P$, since the derivative in J of $\cos(\theta)$ is negative, then its minimum value will occur for the maximum value of J, that is $J = P-1$, since $P > J$. In this case eq. (52) becomes

$$\cos(\theta) = \frac{\sqrt{1+\frac{(P-1)^2}{N-1}}}{\sqrt{P}}. \tag{54}$$

From eqs. (38), (49) and (50), the above value occurs for

$$\boldsymbol{x} = \frac{1}{\sqrt{1+\frac{(P-1)^2}{N-1}}}\left[1 \quad \tfrac{P-1}{N-1} \quad \cdots \quad \tfrac{P-1}{N-1}\right]^{\mathrm{T}}. \tag{55}$$

To find the minimum $\cos(\theta)$ we still have to test the boundary condition $a = 0$. In this case, we have that

$$\|\boldsymbol{x}\|^2 = \sum_{i=1}^{P-J} x^2(i) = 1 \tag{56}$$

$$\text{and } \cos(\theta) = \frac{\sum_{i=1}^{P-J} x(i)}{\sqrt{P}}. \tag{57}$$

and the Lagrangian functional to be minimized is

$$f(x(1),\ldots,x(P-J),\lambda)=\frac{1}{\sqrt{P}}\sum_{i=1}^{P-J}x(i)+\lambda\left(\sum_{i=1}^{P-J}x(i)^2-1\right), \quad (58)$$

and taking derivatives with respect to $x(i)$ we obtain:

$$\frac{\partial f(\cdot)}{\partial x(i)}=0 \;\Rightarrow\; \frac{1}{\sqrt{P}}+2\lambda x(i)=0 \;\Rightarrow\; x(i)=-\frac{1}{2\sqrt{P}\lambda}, \quad (59)$$

and substituting in eq. (56) we have that

$$(P-J)\frac{1}{4P\lambda^2}=1 \quad\Rightarrow\quad x(i)=\frac{1}{\sqrt{P-J}} \quad (60)$$

Therefore, from eq. (57), we have that

$$\cos(\theta)=\frac{\sum_{i=1}^{P-J}\frac{1}{\sqrt{P-J}}}{\sqrt{P}}=\sqrt{\frac{P-J}{P}}. \quad (61)$$

Since $\sqrt{\frac{P-J}{P}}$ then $\cos(\theta)$ in the above equation decreases with J and as $P>J$, then its minimum value occurs for $J=P-1$, that is,

$$\cos(\theta)=\frac{1}{\sqrt{P}}. \quad (62)$$

From eqs. (38) and (60), the above value occurs for $\boldsymbol{x}=\begin{bmatrix}1 & 0 & \cdots & 0\end{bmatrix}^{\mathrm{T}}$.

Then, from eqs. (43), (54) and (62), the minimum value of $\cos(\theta)$ is given by

$$\cos(\Theta(\mathcal{D}))=\min\left\{\sqrt{\frac{P}{N}},\frac{\sqrt{1+\frac{(P-1)^2}{N-1}}}{\sqrt{P}},\frac{1}{\sqrt{P}}\right\}. \quad (63)$$

Since $\sqrt{1+\frac{(P-1)^2}{N-1}}\geq 1$, then one has

$$\cos(\Theta(\mathcal{D}))=\min\left\{\sqrt{\frac{P}{N}},\frac{1}{\sqrt{P}}\right\} \quad (64)$$

Therefore, $\cos(\Theta(\mathcal{D}))$ is given by

$$\cos(\Theta(\mathcal{D}))=\begin{cases}\sqrt{\frac{P}{N}}, & P<\sqrt{N},\\ \frac{1}{\sqrt{P}}, & P\geq\sqrt{N}.\end{cases} \quad (65)$$

B. Codebook choice

The codebook used to perform the approximation of an impulse response $\boldsymbol{h}$ is as described in eq. (20). Then, each step of the MPGBP algorithm adds P powers-of-two to the filter complexity (in fact this is just a lower bound because if we use the *common-term reduction step* in Subsection III-B the complexity tends to be lower). Therefore, if we perform m steps of the MPGBP algorithm, the number of powers-of-two used to implement the filter is bounded by mP.

If we want the approximation error $\|\boldsymbol{h}-\boldsymbol{h}^{(M)}\|$ after M steps to be bounded by ϵ, then, from eq. (14), M should be such that $\|\boldsymbol{h}-\boldsymbol{h}^{(M)}\|\leq\beta^M\|\boldsymbol{h}\|\leq\epsilon$, and thus $M\geq\frac{\log\left(\frac{\epsilon}{\|\boldsymbol{h}\|}\right)}{\log\beta}$. Using the value of β from eq. (24) this relation becomes

$$M\geq\frac{\log\left(\frac{\epsilon}{\|\boldsymbol{h}\|}\right)}{\log\sqrt{1-\frac{8}{9}\cos^2\left(\Theta(\mathcal{D})\right)}}, \quad (66)$$

and thus an upper bound on number of power-of-two becomes

$$N_{\text{POT}}=\left\lceil\frac{2P\log\left(\frac{\epsilon}{\|\boldsymbol{h}\|}\right)}{\log\left[1-\frac{8}{9}\cos^2\left(\Theta(\mathcal{D})\right)\right]}\right\rceil. \quad (67)$$

Above, $\lceil x\rceil$ denotes the smallest integer larger than or equal to $x\in\mathbb{R}$. Substituting $\cos(\Theta(\mathcal{D}))$ from eq. (65) we have

$$N_{\text{POT}}=\begin{cases}\left\lceil\frac{2P\log\left(\frac{\epsilon}{\|\boldsymbol{h}\|}\right)}{\log\left[1-\frac{8}{9}\frac{P}{N}\right]}\right\rceil, & P<\sqrt{N},\\ \left\lceil\frac{2P\log\left(\frac{\epsilon}{\|\boldsymbol{h}\|}\right)}{\log\left[1-\frac{8}{9}\frac{1}{P}\right]}\right\rceil, & P\geq\sqrt{N}.\end{cases} \quad (68)$$

The expression in eq. (68) is a decreasing function of P for $P<\sqrt{N}$ and an increasing function of P for $P\geq\sqrt{N}$. Therefore, its minimum value of N_{POT} occurs for $P=\sqrt{N}$ (in fact, either the smallest integer larger than it or the largest integer smaller than it).

Figure 2 shows a plot of N_{POT} as a function of P for $N=100$. It can be seen that, although the minimum is at $P=\sqrt{N}=10$, the derivative for $P<\sqrt{N}$ is much smaller than the derivative one for $P\geq\sqrt{N}$. This behavior suggests that the number of powers-of-two generated by the algorithm is not to be very sensitive to the value of P provided that $P<\sqrt{N}$.

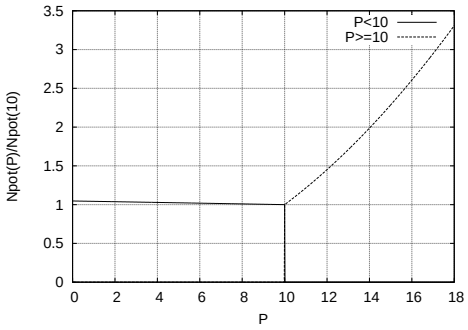

Fig. 2. Number of powers-of-two necessary for a given approximation error in function of P for the dimension N of $\boldsymbol{h}$ equal to 100.

This result tells us that, in the absence of the *common-term reduction step* (Subsection III-B), the optimal N-dimensional codebook should be made up of unit-norm codewords that are permutations of

$$\boldsymbol{c}_j=\left[\pm\left(\frac{1}{\sqrt{P}}\right)^{\lfloor\sqrt{N}\rfloor},0^{N-\lfloor\sqrt{N}\rfloor}\right], \quad (69)$$

where $\lfloor x\rfloor$ denotes the largest integer smaller than or equal to $x\in\mathbb{R}$. However, its performance for values of P smaller than $\lfloor\sqrt{N}\rfloor$ should be similar.

It is important to notice that, besides the common-term reduction, there is another factor that contributes to the number of powers-of-two to be smaller than the figures given by eq. (68). For deducing eq. (68) it was assumed that the angle between the residual and the codebook vector closest to it is equal to $\Theta(\mathcal{D})$ in every iteration, a worst case scenario. Therefore, in practice, the actual number of powers-of-two employed in a filter design tends to be smaller than the one predicted by eq. (68).

V. Complexity Figures

A non-ignored problem of greedy algorithms is their complexity. A priori, if the dictionary has no structure, in a

Matching Pursuit iteration one has to perform a full-search in order to find the codeword closest to a given input vector. In this case, the residue has to be projected into all the dictionary elements to search for the most similar one. Assuming that the signal space dimension is N and that the dictionary has $\#\mathcal{D}$ elements, $\#\mathcal{D}$ inner products of dimension N are necessary, leading to $N\#\mathcal{D}$ multiplications and $(N-1)\#\mathcal{D}$ additions for computing all the inner products and $\#\mathcal{D}\log(\#\mathcal{D})$ comparisons for sorting the inner products values (in average).

Due to this complexity, several strategies have been applied for reducing the computational cost, in general at the expense of memory. For example, in [25] a fast MP algorithm is discussed. In it, instead of computing the inner products between the residue and the dictionary elements at each MP iteration, inner products are computed just in the first MP iteration. Due to the orthogonality between the residue and the selected codeword in each MP iteration, in the subsequent iterations the inner products of the residue and the codewords are updated using the inner products of the selected codeword and the other dictionary elements. In other works, based on the same reasoning (orthogonality between selected codeword and the residue) dictionaries with pre-defined structures favoring complexity reduction have been employed. However, in our case, MPGBP uses in-loop quantization [32]. The coefficients are quantized to the closest power of α, which destroys the orthogonality between the residue and the codeword. This prevents the use of the above mentioned fast algorithms, which tends to contribute to an increased complexity.

In our help comes the fact that, as seen in Section IV in the SOPOT FIR design algorithm here presented, the dictionary elements employed have P elements equal to ± 1 and $N-P$ zeros. This reduces the problem of computing the inner products to sorting the residue coordinates in function of the absolute values and adding the P largest ones. This strategy automatically indicates the P elements of the N-dimensional codeword that must be ± 1, the sign depending on the actual signs of the residue coordinate values. Therefore, in average, each MPGBP iteration has a complexity of $N\log(N)$ comparisons for sorting the residue coordinates. This leads to a fast, efficient algorithm, suitable for hardware implementation.

Algorithm 5: Process to find a set of filters satisfying a prescribed specification. Each candidate uses a bit-depth for coefficients representations of at most b_M. $\gamma < 1$ defines the range of tested filter lengths. It ends up providing a set of possible filter implementations of different lengths N obtained for different P's, having different numbers of POTs and bit-depths. From the generated designs one can choose a specific implementation.
1. Compute the smallest filter length N_m that allows to design a Parks-McClellan filter satisfying the given specification.
2. **For** P = 1 **to** $\left\lfloor \sqrt{N_m(1+\gamma)/2} \right\rfloor$
2.1 **For** N = N_m **to** $\lceil N_m(1+\gamma) \rceil$
2.1.1 $m \leftarrow 1$;
2.1.2 **Set** M with the maximum number of MPGBP iterations
2.1.3 $\boldsymbol{h}_{N,P}^{(1)} \leftarrow$ Parks-McClellan design according to the specifications $F(e^{\jmath\omega})$ and length N
2.1.4 **While** $m \leq M$, apply the MPGBP step to $\boldsymbol{h}_{N,P}^{(m)}$
Obs: This is composed of steps (2.(a)-2.(c)) in Algorithm 2, that have fast implementation using algorithm III. In addition, since $\boldsymbol{h}_{N,P}^{(1)}$ is either symmetric or antisymmetric, the residue length is "half" the filter length.
2.1.4.1 **If** $\boldsymbol{h}_{N,P}^{(m)}$ satisfies the given specification **Then**
Obs: This can be done by computing $\left| H_{N,P}^{(m)}(e^{\jmath\omega}) \right|$ and comparing it against the prescribed specifications
2.1.4.1.1 **Compute** the bit-depth b_h of $\boldsymbol{h}_{N,P}^{(m)}$ and the number of POTs used in $\boldsymbol{h}_{N,P}^{(m)}$;
Obs: The quantity of non-null bits (POTs) in the coefs. $h_{N,P}^{(m)}(n)$ of $\boldsymbol{h}_{N,P}^{(m)}$ is $\sum_{b=1}^{b_h}\sum_{n=0}^{N}\left| h_{N,P}^{(m)}(n,b) \right|$, where for simplicity $h_{N,P}^{(m)}(n,b)$ denotes the b-th bit of coefficient $h_{N,P}^{(m)}(n)$.
2.1.4.1.2 **Save** $\boldsymbol{h}_{N,P}^{(m)}$ as valid SOPOT filter;
2.1.4.2 $m \leftarrow m+1$;

■

The process for obtaining several SOPOT filters satisfying a desired specification is presented in Algorithm 5. One starts with the desired filter specifications, and using a Parks-McClellan filter design approach one finds the minimum filter length N_m required to satisfy teh prescribed specifications. Given N_m one tries filter designs with lengths N within the range N_m and $N_M = \lceil N_m(1+\gamma) \rceil$, where $\gamma \leq 1$ defines a range of filter lengths to be tried. The designed filters are either symmetric or anti-symmetric. In this work we are interested not only in generating filters but also in working out the effect of P (number of non-null elements in the codewords of the dictionary). In order to do so we set tests for P ranging from 1 to $P_M = \left\lfloor \sqrt{N_M/2} \right\rfloor$, in accordance with what has been previously discussed. For these ranges of P and N (N is the tested filter length and $P \in \{1,\ldots,P_M\}$ specifies the dictionary used in the approximation process) one designs filters through the Parks-McClellan optimization and finds their SOPOTs approximations by means of MPGPB

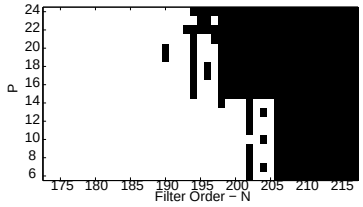

Fig. 3. Success of the filter design for the low-pass filter specifications and algorithm parameters range in Section VI – black → satisfactory design obtained, white → unsatisfactory design obtained.

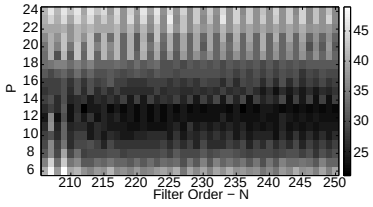

Fig. 4. Iterations employed in the MPGBP design for the filter specifications and algorithm parameters range in Section VI.

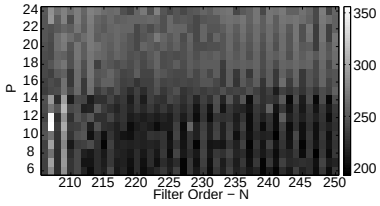

Fig. 5. Adders required for the filters designed using MPGBP for the filter specifications and algorithm parameters range in Section VI.

with dictionaries having P ± 1's and $\lceil N/2 \rceil - P$ zeros[1]. Each approximation is evaluated with respect to the prescribed specifications. One ends with a set of filters $\boldsymbol{h}_{N,P}^{(m)}$ that match the prescribed filter specifications. From this set of filters one may chose a particular design.

For an MPGBP residue of length N', using the dictionaries discussed here, the complexity of the required search for the best matching atom in an MPGBP iteration is on average $N' \log N'$ operations for sorting the residue coefficients – simple operations not requiring any multiplication actually. The residue update requires P operations, updating the P largest coefficients of the residue. Let us assume that at each MPGBP step the filter response is evaluated using an FFT of length $K \gg N$, therefore step 2.1.4.1 in Algorithm 5) requires $K \log_2 K$ (FFT computation) + K (comparisons of the obtained response to the prescribed filter specification) operations. Counting the number of POTs used in $\boldsymbol{h}_{N,P}^{(m)}$ requires at most $N' b_M$ sums as explicitly exemplified of step 2.1.4.1.1 in Algorithm 5, b_M is the largest bit-depth allowed. Therefore, assuming at most M MPGBP steps, and that the filters are either symmetric or anti-symmetric, the number of operations C in Algorithm 5 is bounded by

$$C < M \sum_{P=1}^{P_M} \sum_{N=N_m}^{N_M} \left[(\lceil N/2 \rceil \log \lceil N/2 \rceil) + P + \lceil N/2 \rceil \, b_M + K \log_2 (K) + K \right]. \tag{70}$$

VI. Algorithm Behavior

The proposed algorithm is capable of finding a good sum of powers-of-two representation of a filter prototype in fast way. Being so, the algorithm can be setup for searching for the best filter design by varying the following parameters:

1) The filter length N;
2) Dictionary specification/design, that is the quantity of non-null coordinates P in a given dimension N (which equates the filter length).;
3) The smallest power of two allowed - this impacts on the filter input/output word-length/bit-depth;
4) The total quantity of adders employed in the obtained filter;
5) The total quantity of MP iterations allowed, that restricts the complexity of the design process.

We present some results considering these different aspects. A low-pass filter is considered whose passband and stop-band edges are located at $\omega_p = 0.1\pi$ and $\omega_s = 0.115\pi$ and the maximum allowed ripples are $\delta_p = \delta_s = 0.00316$, i.e. -50dB. The filter prototype has been designed using the Parks-McClellan algorithm [24]. For these specifications, the minimum order leading to a design attaining the specifications is determined to be $N_{\min} = 173$. Different filter are designed with different dictionaries varying the length from 173 up to 250, with maximum bit-depth of 32 bits, allowing at most 1000 adders, and at most 400 MP iterations. The dictionaries are composed of atoms having P out of N coefficients equal to ± 1, for different values of P, spanning from 6 to 24.

Fig. 3 shows for which values of P and N a satisfactory filter design is achieved for the specifications and value ranges discussed above. The black region indicates a successful filter design, i.e., the algorithm obtains a filter design satisfying the filter specifications within the restrictions in parameter ranges. The white region indicates unsuccessful filter design. As can be observed there is neighbourhood between a region where no satisfactory filter is obtained to the one in which satisfactory filters (i.e., that satisfy the specifications) are always obtained. The values of N greater than 217 have been omitted from Fig. 3 since they always lead to a successful design.

To better understand the algorithm behaviour we measure some of the parameters of the designed filters in a region where a satisfactory design is always obtained ($N \geq 206$).

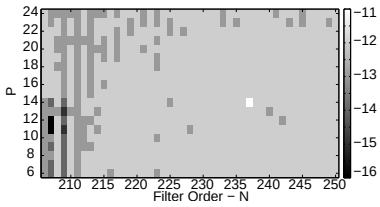

Fig. 6. Smallest power-of-two required for the filters designed for the specifications and MPGBP algorithm parameters range in Section VI. The higher the magnitude of the smallest power-of-two, the larger the word length necessary for implementing the filter.

[1]Note that the dictionaries are defined considering half the filter length or half the filter length plus 1 as dimension, due to their symmetric or anti-symmetric nature.

Fig. 4 shows how the quantity of MP iterations employed in the design vary with P and N. This figure shows that the value of P that gives the smallest number of iterations is around $\sqrt{N}$. This is in agreement with eqs. (65) and (66), from which one can deduce that the smallest number of iterations M is obtained for $P = \sqrt{N}$. We can notice that, given N, less iterations tend to be employed when P is close to and smaller than $\sqrt{N}$. Fig. 5. shows the variation of the number of adders (powers-of-two) of the designed filters with P and N. This corroborates the result in presented in eq. (68) and Fig. 2. It says that, for a given approximation error, the value of P that minimizes the necessary number of powers-of-two, if no common-terms reduction is employed, is $P = \lfloor\sqrt{N}\rfloor$, but smaller values of P yield similar performance. Again, it is important to notice that this result might not correspond exactly to what is observed in the experiment due to two main reasons. The first one is that, in our results, the common-term reduction step, as described in Subsection III-B, is indeed employed. In addition, eq. (68) supposes that the angle between the residual to be coded and the closest codeword is $\Theta(\mathcal{D})$ in every iteration; this is a worst case, and thus provides only a lower bound on performance. However, as can be seen from Fig. 5, eq. (68) gives a good indication of the algorithm's performance. Fig. 6 concludes this experiment by evaluating the word length necessary for the filter implementation. It shows the smallest power-of-two used in the designed filter, since the smaller the magnitude of the smallest power-of-two, the longer the word length.

VII. Design Examples

In this section we compare the performance of the proposed algorithm against that of other methods in the literature for the design of SOPOTs filters. The filter prototypes for our design[2] are obtained using the Parks-McClellan algorithm [24].

1) Example 1 – Low Pass Filter [4,7,14]: Consider the specifications in [7], Example 1, also in [14] and [4]. A low-pass filter is desired in which the passband and stop-band edges are located at $\omega_p = 0.3\pi$ and $\omega_s = 0.5\pi$ and the maximum allowed passband and stop-band ripples are $\delta_p = \delta_s = 0.00316$, corresponding to a 50 dB stop-band attenuation. For the Parks-McClellan prototype design, the minimum required order is $N_{\min} = 25$. Since the filters are symmetric it suffices to apply the MPGBP method at half of its coefficients. We employ the design method for orders up to $N = 37$ and therefore a search space of at most $N' = 18$ coefficients, and values of $P \in \{1, 2, 3, 4\}$, searching for the minimal implementation complexity (in terms of number of sum of powers-of-two/adders).

The best results presented in [7] indicate that the specifications can be attained by using an $N = 29$ order filter, with 37 adders. They argue that the number of adders can be lowered by using a common sub-expression elimination post-design method. In our results we do not consider the application of this sub-expression elimination method since, although it might reduce the number of adders, it does not help to grasp the proposed method behavior. As can be seen from Table III, the proposed method consistently obtains designs not only comparable but with lower complexities to the one in [7], using a minimum of 28 adders for a filter of length 33, while for a filter of length 29, 34 adders are required. The amplitude responses for two MPGBP-designed filters, along with their passband details, can be seen in Fig. 7 and 8. The impulse response of the particular filter obtained with $N = 33$ and $P = 4$ is presented in Table IV.

[2]Software implementing the examples presented is available at: http://www.prosaico.uerj.br/MPGBP_filter_approx.tgz

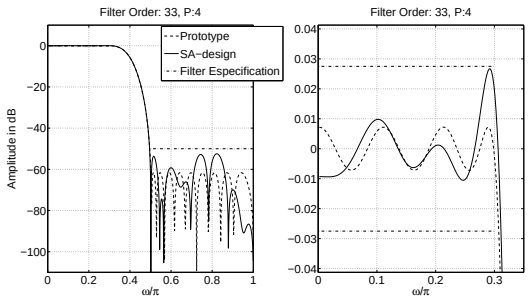

Fig. 7. Magnitude responses for the prototype and the SA-designed FIR filters ($N = 33$ and $P = 4$) in Example 1. (Passband detail shown separately)

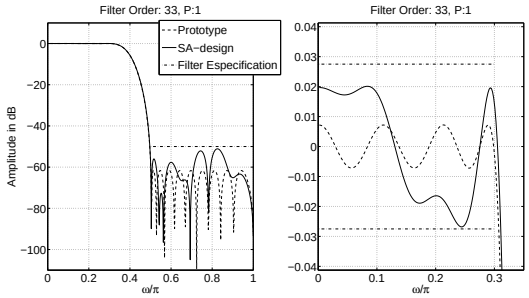

Fig. 8. Magnitude responses for the prototype and the SA-designed FIR filters ($N = 33$ and $P = 1$) in Example 1.(Passband detail shown separately)

In order to compare some characteristics of the designed filters we employ the Normalized Peak Ripple (NPR) as in [14]. For the specifications considered, in [14] the NPR are -53.64 and -51.56 dB for filters of lengths 33 and 35 respectively using an average of 2 adders per coefficient. As can be seen in the last two columns in Table III the NPR for the filters obtained using the proposed approach

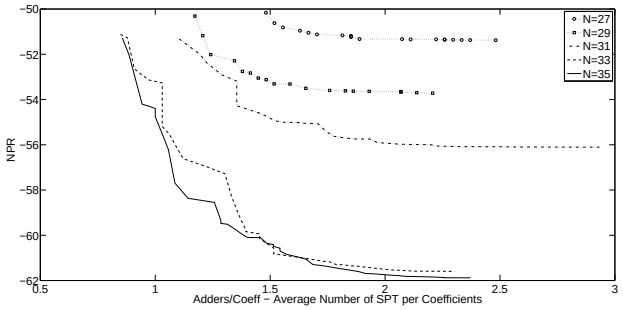

Fig. 9. Normalized Peak Ripple (dB) in function of the quantity of adders per coefficient (Adders/Coeff) for the filter with specifications of Example 1, considering filter lengths in $\{27, 29, 31, 33, 35\}$.

TABLE III
IMPLEMENTATION PARAMETERS FOR SPECIFICATIONS OF EXAMPLE 1.

Adders	N	P	POTs – bit-depth	MPGBP Iters.	Adders/Coeff	NPR (dB)
28	33	1	11	28	.8485	-51.12
30	35	1	11	30	.8571	-51.29
31	33	3	11	12	.9394	-52.46
31	35	3	11	12	.8857	-52.01
31	33	4	11	11	.9394	-52.48
31	35	4	11	11	.8857	-51.72
32	33	2	11	17	.9697	-53.13
32	35	2	11	17	.9143	-51.80
34	29	1	12	34	1.1724	-50.31
34	31	1	12	34	1.0968	-51.27
34	31	2	11	18	1.0968	-50.84
34	31	4	11	12	1.0968	-50.84

TABLE IV
COEFFICIENTS FOR THE SA-DESIGNED FIR FILTER OF EXAMPLE 1.
($P = 4,\ N = 33, h(33-n) = h(n)$ for $n = 0, \ldots, 15$)

n	$h(n)$	POTs	n	$h(n)$	POTs
0	0.0009765625	2^{-10}	9	0.0	0
1	0.0009765625	2^{-10}	10	0.033203125	$2^{-5}+2^{-9}$
2	-0.001953125	-2^{-9}	11	0.02685546875	$2^{-5}-2^{-8}-2^{-11}$
3	-0.00390625	-2^{-8}	12	-0.03564453125	$-2^{-5}-2^{-8}-2^{-11}$
4	0.0	0	13	-0.07861328125	$2^{-4}-2^{-6}-2^{-11}$
5	0.00830078125	$2^{-7}+2^{-11}$	14	0.0	0
6	0.0068359375	$2^{-7}-2^{-10}$	15	0.1982421875	$2^{-2}-2^{-4}+2^{-7}+2^{-8}-2^{-10}$
7	-0.009765625	$-2^{-7}-2^{-9}$	16	0.373046875	$2^{-1}-2^{-3}-2^{-9}$
8	-0.01953125	$-2^{-6}-2^{-8}$			

are comparable to those in [14] using less Adders/Coeff (quantity of adders divided by the quantity of coefficients of the SOPOT approximation). In order to further investigate the NPR of the filters designed using the proposed approach, we present in Figure 9 a graph of the best attainable NPR as a function of Adders/Coeff for filters designed accordingly to the specifications in this example. For obtaining these results we consider several approximations of the filters of length N for a given dictionary and a varying number of decomposition steps not looking for the smallest complexity (quantity of adders). This graph can be compared to the one in Figure 2 of [14], there NPR starts around -40dB for an average number of Adders/Coeff of 1, it is easy to see that our results present a much better performance in terms of this metric.

The proposed algorithm was implemented in Matlab, for the 48 filters ($N \in \{25, 26, \ldots, 36, 37\}$ and $P \in \{1, 2, 3, 4\}$) filter approximations the Average CPU time / Filter Approximation is 0.076274 (secs) running on a KUbuntu 64bits Operating System on a Intel(R) Core(TM) i7-3537U CPU 2.00GHz. If one considers just the design approximations that satisfy the filter requirements this time drops to 0.058603, what considers in 44 of the 48 cases. It should be pointed that these times consider the evaluations by means of FFT computations (of length $K = 2048$ in this example) and the Parks-McClellan prototypes generation. Figure 10 tries to summarize the required computing time by presenting the histogram of the times elapsed in the filter approximations considered. The graph at the top presents the histogram of the design times (prototype + MPGBP iters + Frequency Response Evaluation carried at each MPGBP step) for the 48 filters, while the graph at the bottom presents the same metric for the filters (N,P for varying quantities of MPGPB decomposition steps) satisfying the filter specifications.

2) Example 2 – Low Pass Filter [19,35]: This example employs the specification considered in [19,35]. A low-pass

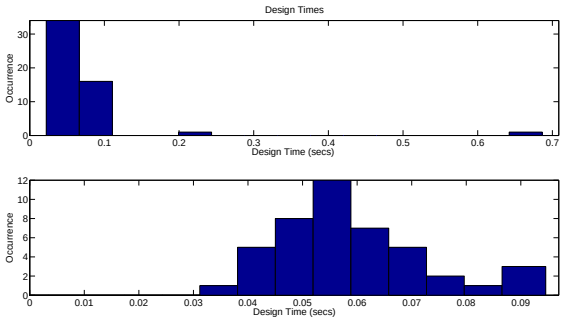

Fig. 10. Histogram of computing times (prototype design + MPGBP + filter response evaluation (at each MPGBP step) for obtaining filters satisfying the specifications in Example 1. The histogram on the top consider successful and unsuccessful approximation cases, while the graph on the bottom considers just the satisfactory ones. These are obtained running the Matlab algorithm on a KUbuntu 64bits Operating System on a Intel(R) Core(TM) i7-3537U CPU 2.00GHz.

TABLE V
IMPLEMENTATION PARAMETERS FOR SPECIFICATIONS OF EXAMPLE 2.

Adders	N	P	POTs – bit-depth	MPGBP Iters.
73	51	2	11	37
75	51	1	11	75
77	51	3	11	26
79	51	4	11	21
80	54	5	11	17
82	51	5	12	17
83	52	1	11	83
83	56	1	11	83
83	52	2	11	42

filter is desired in whose passband and stop-band edges are located at $\omega_p = 0.042\pi$ and $\omega_s = 0.14\pi$ and the maximum allowed passband and stop-band ripples are $\delta_p = .2dB$ and $\delta_s = -60dB$. For the Parks-McClellan prototype design, the minimum required order is $N_{\min} = 46$. We investigate designs using the proposed approach for $P \in \{46, \ldots, 61\}$ with dictionaries defined by $P \in \{1, 2, 3, 5\}$, since $\lceil \sqrt{61/2} \rceil = 5$. Our shortest filter satisfying the specifications is obtained for $N = 50$ using 84 adders and the one using the least number of adders has length 51 using 73 adders. In [19] this filter is design using 59 taps and 71 adders while in [35] a 60 tap filter is obtained using 87 adders. A sample of the parameters of satisfactory filters designed is presented in Table V ordered in function of the quantity of adders employed in the filter implementation. The coefficients of the smaller complexity filter are presented in Table VI. The frequency response of this filter is presented in Figure 11 and the coefficients of the original filter and the approximation obtained, together with the approximation error, are presented in Figure 12.

3) Example 3 – Band Pass Filter : In order to evaluate the proposed approach for non low-pass filters, we propose the design of a band-pass filter whose specifications are:

- passband edges: $\omega_{p1} = 0.3\pi$, $\omega_{p2} = 0.6\pi$,
- stop-band edges: $\omega_{s1} = 0.15\pi$, $\omega_{s2} = 0.8\pi$
- and the maximum allowed passband and stop-band ripples are $\delta_p = \delta_{s1} = \delta_{s2} = 0.001$, corresponding to a 60 dB stop-band attenuation.

For the prototype design, the minimum required order leading to a design attaining the specifications is determined to be $N_{\min} = 43$. We employ the design method for values of

TABLE VI
COEFFICIENTS FOR THE SA-DESIGNED FIR FILTER OF EXAMPLE 2.
($P = 2,\ N = 51, h(51-n) = h(n)$ for $n = 0, \ldots, 26$)

n		POTs	n	POTs
1	$-2^{-10} + 2^{-12}$	14	-2^{-14}	
2	$-2^{-10} - 2^{-13}$	15	$2^{-8} + 2^{-10}$	
3	-2^{-9}	16	$2^{-6} - 2^{-8} - 2^{-11} - 2^{-13}$	
4	$-2^{-8} + 2^{-10} + 2^{-13}$	17	$2^{-6} + 2^{-8} - 2^{-10} - 2^{-13}$	
5	-2^{-8}	18	$2^{-5} - 2^{-8} - 2^{-10} + 2^{-12}$	
6	$-2^{-8} - 2^{-10} - 2^{-12}$	19	$2^{-5} + 2^{-8} + 2^{-12}$	
7	$-2^{-7} + 2^{-9} - 2^{-11}$	20	$2^{-4} - 2^{-6} - 2^{-9} - 2^{-11}$	
8	$-2^{-7} + 2^{-11}$	21	$2^{-4} - 2^{-7} - 2^{-10} - 2^{-11} + 2^{-13}$	
9	$-2^{-7} - 2^{-12}$	22	$2^{-4} - 2^{-10} + 2^{-14}$	
10	$-2^{-7} - 2^{-11}$	23	$2^{-4} + 2^{-7} - 2^{-9} - 2^{-11}$	
11	-2^{-7}	24	$2^{-4} + 2^{-6} - 2^{-8} + 2^{-11} - 2^{-14}$	
12	$-2^{-7} + 2^{-9} + 2^{-11}$	25	$2^{-4} + 2^{-6} + 2^{-11} + 2^{-13}$	
13	$-2^{-8} + 2^{-13}$	26	$2^{-4} + 2^{-6} + 2^{-9} + 2^{-10} - 2^{-12}$	

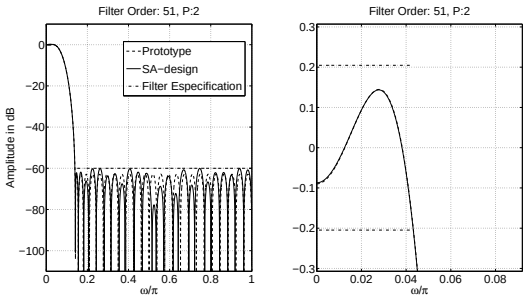

Fig. 11. Magnitude responses for the prototype and the SA-designed FIR filters ($N = 51$ and $P = 2$) in Example 2. (Passband detail shown separately)

$P \in \{1, 2, 3, 4, 5\}$ and order up to $N = 64$, searching for the minimum implementation complexity in terms of number of sum of powers-of-two/adders.

The implementation with lowest complexity is obtained by setting $P = 1$ and designing an $N = 52$ order filter. For this design, whose coefficients are given in Table VIII, the number of required adders is 66, with the use of 12 different power-of-two factors. The amplitude response for the resulting filter, along with its passband details, can be seen in Fig. 13.

Figure 14 presents the times required for computing the filters in this example. As the filters are longer than the ones in Example 1 the computing times increase. In this case from the 110 designs tried, 100 provide an implementation satisfying the filter specifications.

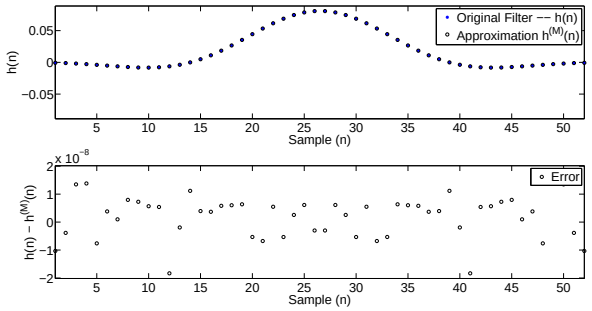

Fig. 12. Prototype and the SA-designed FIR filters samples ($N = 51$ and $P = 2$) in Example 2. Approximation error (differences between samples of the original and the approximated filter) is shown at the bottom.

TABLE VII
IMPLEMENTATION PARAMETERS FOR SPECIFICATIONS OF EXAMPLE 3.

Adders	N	P	POTs – bit-depth	MPGBP Iters.
66	52	1	12	66
67	55	4	12	18
68	49	1	11	68
68	49	2	11	34
69	58	1	11	69
69	52	2	13	37
69	52	4	12	20
69	49	5	11	16
70	49	4	12	20
70	52	5	12	16

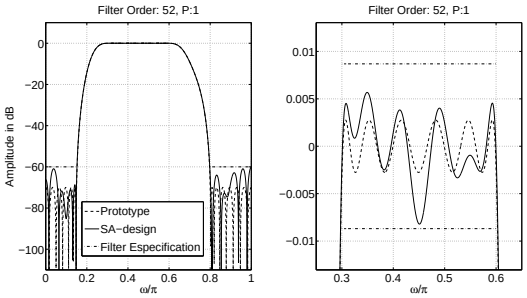

Fig. 13. Magnitude responses for the prototype and the SA-designed FIR filters ($N = 52$ and $P = 1$) in Example 3. (Passband detail shown separately)

TABLE VIII
COEFFICIENTS FOR THE SA-DESIGNED FIR FILTER OF EXAMPLE 3.
($N = 52,\ P = 1,\ h(51-n) = h(n)$ for $n = 0, \ldots, 25$)

n	$h(n)$	POTs	n	$h(n)$	POTs
0	-0.000244140625	-2^{-12}	14	-0.00732421875	$-2^{-7}+2^{-11}$
1	0.0009765625	2^{-10}	15	-0.029541015625	$-2^{-5}+2^{-9}-2^{-12}$
2	0.0008544921875	$2^{-10}-2^{-13}$	16	-0.0078125	-2^{-7}
3	-0.0001220703125	-2^{-13}	17	0.00732421875	$2^{-7}-2^{-11}$
4	0.000732421875	$2^{-10}-2^{-12}$	18	-0.006591796875	$-2^{-7}+2^{-10}+2^{-12}$
5	-0.00146484375	$-2^{-9}+2^{-11}$	19	0.0548095703125	$2^{-4}-2^{-7}+2^{-13}$
6	-0.004150390625	$2^{-8}-2^{-12}$	20	0.0562744140625	$2^{-4}-2^{-7}+2^{-9}-2^{-11}+2^{-13}$
7	-0.0009765625	-2^{-10}	21	-0.0297851562 5	$-2^{-5}+2^{-9}-2^{-11}$
8	-0.00048828125	-2^{-11}	22	0.02685546875	$2^{-5}-2^{-8}-2^{-11}$
9	0.000244140625	2^{-12}	23	-0.067626953125	$-2^{-4}-2^{-8}-2^{-10}-2^{-12}$
10	0.0087890625	$2^{-7}+2^{-10}$	24	-0.2978515625	$-2^{-2}-2^{-4}+2^{-6}-2^{-10}$
11	0.0093994140625	$2^{-7}+2^{-9}-2^{-11}+2^{-13}$	25	0.056396484375	$2^{-4}-2^{-7}+2^{-9}-2^{-12}$
12	-0.000732421875	$-2^{-10}+2^{-12}$	26	0.463134765625	$2^{-1}-2^{-5}-2^{-7}+2^{-9}+2^{-12}$
13	0.000732421875	$2^{-10}-2^{-12}$			

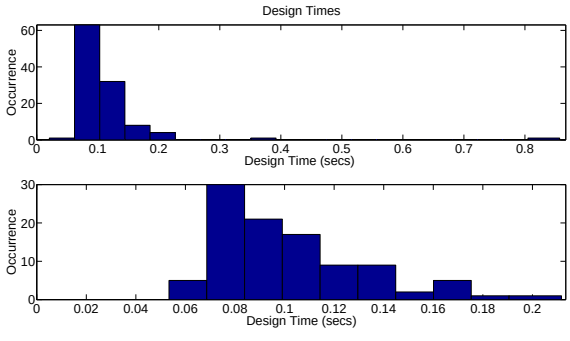

Fig. 14. Histogram of computing times (prototype design + MPGBP + filter response evaluation (at each MPGBP step) for obtaining filters satisfying the specifications in Example 3. The histogram on top consider successful and unsuccessful approximation cases, while the graph at the bottom considers just the satisfactory ones. These are obtained running the Matlab algorithm on a KUbuntu 64bits Operating System on a Intel(R) Core(TM) i7-3537U CPU 2.00GHz.

VIII. Conclusions

We have presented a novel method for the design of Sum of Powers of Two (SOPOT) FIR digital filters. The presented method attempts to minimize the number of sums of powers-of-two employed in the filter implementation. It is based on the decomposition of their impulse response in generalized bitplanes. The decomposition in generalized bitplanes employs successive approximation of the impulse response using vectors from a dictionary. The dictionary used for filter design has only vectors with coefficients equal to $+1$, -1 or 0, and its elements are weighted by powers-of-two when approximating an impulse response. The decomposition is made using the Matching Pursuits in Generalized Bitplanes (MPGBP) algorithm, that has been initially developed for video compression applications. A novel variation of the MPGBP algorithm has been proposed, and a Theorem giving bounds for its performance has been proved. This theorem states that the approximation performance of the dictionaries in the MPGBP algorithm is determined by $\Theta(\mathcal{D})$, the largest angle between any vector in $\mathbb{R}^N$ and its closest vector in the dictionary. In this paper we also give an analytical expression for the value of $\Theta(\mathcal{D})$ for the codebooks used, leading to the choice of optimal codebooks for a given filter order. Unlike its counterparts, that resort to costly optimization techniques, the proposed algorithm has extremely low computational complexity. Yet, experimental results show that the designs generated give results similar to the best ones in the literature. In addition, the proposed approach is much faster to design the rather sophisticate filters and requires no sophisticate setup of initial parameters. As seen, it can be employed to efficiently approximate different filters responses satisfying different passband and stop-band specifications of any order, therefore one can employ it to design filters matching different implementation criteria as quantity of SOPOTs, bit-depth, length.

References

[1] H. Samueli, "An improved search algorithm for the design of multiplierless fir filters with powers-of-two coefficients," *Circuits and Systems, IEEE Transactions on*, vol. 36, pp. 1044 –1047, jul. 1989.

[2] J. Evans, "Efficient fir filter architectures suitable for fpga implementation," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, vol. 41, pp. 490 –493, jul. 1994.

[3] M. Macleod and A. Dempster, "Multiplierless fir filter design algorithms," *Signal Processing Letters, IEEE*, vol. 12, pp. 186 – 189, march 2005.

[4] Y. C. Lim, R. Yang, D. Li, and J. Song, "Signed power-of-two term allocation scheme for the design of digital filters," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, vol. 46, pp. 577 –584, may 1999.

[5] K. Yeung and S. Chan, "Multiplier-less fir digital filters using programmable sum-of-power-of-two (sopot) coefficients," in *Field-Programmable Technology, 2002. (FPT). Proceedings. 2002 IEEE International Conference on*, pp. 78 – 84, dec. 2002.

[6] Y. J. Yu, T. Saramaki, and Y. C. Lim, "An iterative method for optimizing fir filters synthesized using the two-stage frequency-response masking technique," in *Circuits and Systems, 2003. ISCAS '03. Proceedings of the 2003 International Symposium on*, vol. 3, pp. III–874 – III–877 vol.3, may 2003.

[7] J. Yli-Kaakinen and T. Saramaki, "A systematic algorithm for the design of multiplierless FIR filters," in *Circuits and Systems, 2001. ISCAS 2001. The 2001 IEEE International Symposium on*, vol. 2, pp. 185–188 vol. 2, 6-9 2001.

[8] R. Cemes and D. Ait-Boudaoud, "Genetic approach to design of multiplierless fir filters," *Electronics Letters*, vol. 29, pp. 2090 –2091, nov. 1993.

[9] Y. J. Yu and Y. C. Lim, "Design of linear phase FIR filters in subexpression space using mixed integer linear programming," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 54, pp. 2330–2338, oct. 2007.

[10] J. Yli-Kaakinen and T. Saramaki, "A systematic algorithm for the design of lattice wave digital filters with short-coefficient wordlength," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 54, pp. 1838–1851, aug. 2007.

[11] Y. C. Lim, S. Parker, and A. Constantinides, "Finite word length FIR filter design using integer programming over a discrete coefficient space," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 30, pp. 661–664, aug. 1982.

[12] Y. C. Lim and S. Parker, "FIR filter design over a discrete powers-of-two coefficient space," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 31, pp. 583–591, jun. 1983.

[13] P. Gentili, F. Piazza, and A. Uncini, "Efficient genetic algorithm design for power-of-two fir filters," in *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 2, pp. 1268 –1271 vol.2, may 1995.

[14] D. Li, Y. C. Lim, Y. Lian, and J. Song, "A polynomial-time algorithm for designing fir filters with power-of-two coefficients," *Signal Processing, IEEE Transactions on*, vol. 50, pp. 1935 –1941, aug. 2002.

[15] F. Brglez, "Digital filter design with short word-length coefficients," *Circuits and Systems, IEEE Transactions on*, vol. 25, no. 12, pp. 1044–1050, 1978.

[16] D. M. Kodek, "Design of optimal finite wordlength fir digital filters using integer programming techniques," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 3, pp. 304–308, 1980.

[17] D. Kodek and K. Steiglitz, "Filter-length word-length tradeoffs in fir digital filter design," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 6, pp. 739–744, 1980.

[18] D. M. Kodek, "Performance limit of finite wordlength fir digital filters," *Signal Processing, IEEE Transactions on*, vol. 53, no. 7, pp. 2462–2469, 2005.

[19] J. Skaf and S. P. Boyd, "Filter design with low complexity coefficients," *Signal Processing, IEEE Transactions on*, vol. 56, no. 7, pp. 3162–3169, 2008.

[20] Y. Cao, K. Wang, W. Pei, Y. Liu, and Y. Zhang, "Design of high-order extrapolated impulse response fir filters with signed powers-of-two coefficients," *Circuits, Systems, and Signal Processing*, vol. 30, no. 5, pp. 963–985, 2011.

[21] R. Caetano, E. A. B. da Silva, and A. G. Ciancio, "Video coding using greedy decompositions on generalised bit-planes," *Electronics Letters*, vol. 38, pp. 507–508, 2002.

[22] S. Mallat, *A Wavelet Tour of Signal Processing, Second Edition (Wavelet Analysis & Its Applications)*. Academic Press, 2nd ed., 1999.

[23] K. Sayood, *Introduction to Data Compression*. Morgan Kaufmann, 2nd ed., 2000.

[24] P. S. R. Diniz, E. A. B. da Silva, and S. L. Netto, *Digital Signal Processing: System Analysis and Design*. Cambridge University Press, 2nd ed., 2010.

[25] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *Signal Processing, IEEE Transactions on*, vol. 41, pp. 3397–3415, dec 1993.

[26] W. Rudin, *Real and Complex Analysis*. McGraw-Hill, May 1986.

[27] F. Bergeaud and S. Mallat, "Matching pursuit of images," in *Time-Frequency and Time-Scale Analysis, 1994., Proceedings of the IEEE-SP International Symposium on*, pp. 330–333, oct 1994.

[28] G. M. Davis, S. G. Mallat, and Z. Zhang, "Adaptive time-frequency decompositions," *Optical Engineering*, vol. 33, no. 7, pp. 2183–2191, 1994.

[29] G. Davis, S. Mallat, and M. Avellaneda, "Adaptive greedy approximations," *Constructive approximation*, vol. 13, no. 1, pp. 57–98, 1997.

[30] R. A. DeVore and V. N. Temlyakov, "Some remarks on greedy algorithms," *Advances in computational Mathematics*, vol. 5, no. 1, pp. 173–187, 1996.

[31] R. Gribonval and P. Vandergheynst, "On the exponential convergence of matching pursuits in quasi-incoherent dictionaries," *Information Theory, IEEE Transactions on*, vol. 52, no. 1, pp. 255–261, 2006.

[32] L. Lovisolo, E. A. da Silva, and P. S. Diniz, "On the statistics of matching pursuit angles," *Signal Processing*, vol. 90, no. 12, pp. 3164 – 3184, 2010.

[33] L. Lovisolo and E. A. B. da Silva, "Uniform distribution of points on a hyper-sphere with applications to vector bit-plane encoding," *Vision, Image and Signal Processing, IEE Proceedings -*, vol. 148, pp. 187–193, 2001.

[34] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*. Wiley, 2nd ed., 1993.
[35] F. Xu, C. H. Chang, and C. C. Jong, "Design of low-complexity fir filters based on signed-powers-of-two coefficients with reusable common subexpressions," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 26, no. 10, pp. 1898–1907, 2007.
[36] C. S. Ogilvy, *Excursions in Geometry*. Oxford University Press, 1969.

APPENDIX A
PROOF OF MPGBP CONVERGENCE

Theorem 1 (MPGBP Convergence): Given a dictionary $\mathcal{D}$ such that if $\boldsymbol{c}_j \in \mathcal{D}$ then necessarily $-\boldsymbol{c}_j \in \mathcal{D}$ and also $\|\boldsymbol{c}_j\| = 1$, then for $\alpha < 1$ and any input vector $\boldsymbol{x}$ the MPGBP Algorithm (Algorithm 2) converges, and the error incurred in the approximation $\boldsymbol{x}^{(\widetilde{M})}$ is bounded by

$$\left\|\boldsymbol{x} - \boldsymbol{x}^{(M)}\right\|^2 \leq \beta^M \|\boldsymbol{x}\|, \tag{71}$$

$$\text{where } \beta = \sqrt{1 - \frac{4\alpha}{(1+\alpha)^2}\cos^2\left(\Theta(\mathcal{D})\right)}, \tag{72}$$

$$\text{and } \Theta(\mathcal{D}) = \arccos\left\{\min_{\boldsymbol{x}\in\mathbb{R}^N}\left\{\max_{\boldsymbol{c}\in\mathcal{D}}\left(\frac{<\boldsymbol{x},\boldsymbol{c}>}{\|\boldsymbol{x}\|\,\|\boldsymbol{c}\|}\right)\right\}\right\}. \tag{73}$$

Proof: If we can prove that $\exists\beta < 1$ such that the residuals at steps m and $m+1$ follow the constraint

$$\beta_m = \frac{\|\boldsymbol{r}_{m+1}\|}{\|\boldsymbol{r}_m\|} \leq \beta, \quad \forall\, m \geq 1, m \in \mathbb{N}, \tag{74}$$

$$\text{then we have that } \|\boldsymbol{r}_{M+1}\| \leq \|\boldsymbol{r}_1\|\beta^M. \tag{75}$$

Since $\boldsymbol{r}_1 = \boldsymbol{x}$ and $\boldsymbol{r}_{M+1} = \boldsymbol{x} - \boldsymbol{x}^{(M)}$, the above equation would imply eq. (71). Therefore, in order to prove Theorem 1 it suffices to prove eq. (74) and to show that the minimum value of β for which eq. (74) is valid is given by eq. (72).

The argument for this proof is based on Fig. 15, where we have the following correspondences:

a.1) The vector to be approximated at step m, $\boldsymbol{r}_m$, is given by $\overrightarrow{OP}$.
a.2) The direction of the unit-norm vector $\boldsymbol{c}_{i_m} \in \mathcal{D}$, used to approximate $\boldsymbol{r}_m$, is the one of $\overrightarrow{OE}$.
a.3) The coefficient of $\boldsymbol{c}_{i_m}$, that is, the magnitude of the approximation of the residue at step m, is given by the length of $\overline{OE}$.
a.4) The residue $\boldsymbol{r}_{m+1}$ is given by $\overrightarrow{EP}$.
a.5) The position of the points F, E', D, E, C and E'' is based on the fact that $\alpha < 1$.

From the above correspondences, we see that β_m in eq. (74) above is equal to $\frac{\overline{PE}}{\overline{PO}}$. Therefore, the loci of points P such that β_m is constant is the circle of Appolonius [36] for which the ratio of the distance to the focus O to the distance to the focus E is equal to β_m. This circle is illustrated in Fig. 15 as the dashed circumference passing through point P.

In order to carry out the proof of Theorem 1, we will use some properties of Appolonian circles that are described with reference to Fig. 16, that shows circles of Appolonius of foci A and B and several ratios δ of the distance to B to the distance to A. The following properties should be noted [36]:

b.1) For $\delta = 1$ the circle of Appolonius has infinite radius, and is the bisector of segment $\overline{AB}$.
b.2) For $\delta = 0$ the circle of Appolonius is identical to point B. As δ increases up to 1, its radius increases but it remains contained in half-plane to the right of the bisector of $\overline{AB}$.
b.3) For $\delta > 1$ the circle of Appolonius in contained in the half-plane to the left of the bisector of $\overline{AB}$. As δ increases, its radius decreases, and for $\delta \to \infty$ it shrinks to point A.
b.4) The circles of Appolonius for different values of δ never intersect.

If the point P' in Fig. 15 is the projection of P onto $\overline{OE}$, we have that $\overline{OP'}$ is equal to $\langle \boldsymbol{r}_m, \boldsymbol{c}_{j_m} \rangle$ (note that this value is always positive because $\mathcal{D}$ is such that both $\boldsymbol{c}_i$ and $-\boldsymbol{c}_i$ belong to $\mathcal{D}$, for every i).

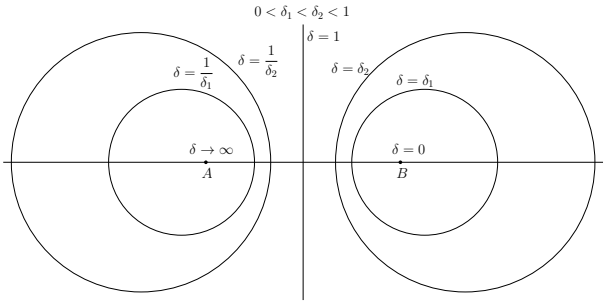

Fig. 16. Circles of Appolonius of foci A and B and several ratios δ of the distance to B to the distance to A.

Therefore, if $\boldsymbol{r}_m$ is approximated with coefficient α^{k_m}, then, from eq. (10), we have that

$$\frac{\alpha^{k_m+1} + \alpha^{k_m}}{2} \leq \overline{OP'} < \frac{\alpha^{k_m} + \alpha^{k_m-1}}{2}. \tag{76}$$

Referring to Fig. 15, this is equivalent to saying that P is closer to E than it is to both E' and E''. Eq. (76) implies that point P is contained in the region between the lines $\overline{AD}$ and $\overline{BC}$. In addition, since the angle between $\boldsymbol{r}_m$ and $\boldsymbol{c}_{i_m}$ is smaller than $\Theta(\mathcal{D})$, we have that P is contained within the quadrilateral $ABCD$.

If we define F as the intersection of the bisector of segment $\overline{OE}$, and segment $\overline{OE}$, as $\overline{OE} = \alpha^{k_m}$, we have that $\overline{OF} = \frac{1}{2}\alpha^{k_m}$. Since

$$\frac{\alpha^{k_m}}{2} < \frac{\alpha^{k_m} + \alpha^{k_m+1}}{2}, \tag{77}$$

and P is inside $ABCD$ therefore P belongs to the half-plane to the right of the bisector of segment $\overline{OE}$. Thus, it belongs to an Appolonian circle of foci O and E that is in the half-plane to the right of the bisector of $\overline{OE}$. From Fig. 16 this is equivalent to saying that $\beta_m = \frac{\overline{PE}}{\overline{PO}} < 1$, $\forall m$. Using eqs. (74) and (75) this proves the convergence of the MPGBP algorithm, since it shows that the residue magnitude goes to zero as the number of steps m goes to infinity.

Now, in order to complete the proof, we have to show that β in eq. (72) is an upper limit for the β_m. We can do this based on two properties demonstrated above:

c.1) P belongs to a circle of Appolonius to the right of the bisector of $\overline{OE}$. This implies that the larger β_m is, the larger is the radius of the circle of Appolonius it belongs to (see Fig. 16).
c.2) P is contained in the quadrilateral $ABCD$.

Property c.2 implies that the Appolonius circle of largest radius satisfying property c.1 will necessarily pass through one of the vertexes of $ABCD$. Therefore, an upper limit for β_m will be the maximum among the β_m for the vertexes A, B, C and D.

- For point A: $\beta_A = \frac{\overline{AE}}{\overline{AO}} = \sqrt{1 - \frac{4\alpha}{(1+\alpha)^2}\cos^2\left(\Theta(\mathcal{D})\right)}$, (78)

$$\text{since } \overline{AD} = \frac{\alpha^{k_m} + \alpha^{k_m+1}}{2}\tan\left(\Theta(\mathcal{D})\right) \tag{79}$$

$$\overline{DE} = \frac{\alpha^{k_m} - \alpha^{k_m+1}}{2} \tag{80}$$

$$\overline{AE}^2 = \overline{AD}^2 + \overline{DE}^2 = \left(\frac{\alpha^{k_m} + \alpha^{k_m+1}}{2}\right)^2 \tan^2\left(\Theta(\mathcal{D})\right) + $$
$$+ \left(\frac{\alpha^{k_m} - \alpha^{k_m+1}}{2}\right)^2 = \frac{\alpha^{2k_m}}{4}\left[(1+\alpha)^2\sec^2\left(\Theta(\mathcal{D})\right) - 4\alpha\right]$$
$$\overline{AO} = \frac{\alpha^{k_m} + \alpha^{k_m+1}}{2}\sec\left(\Theta(\mathcal{D})\right) = \frac{\alpha^{k_m}}{2}(1+\alpha)\sec\left(\Theta(\mathcal{D})\right).$$

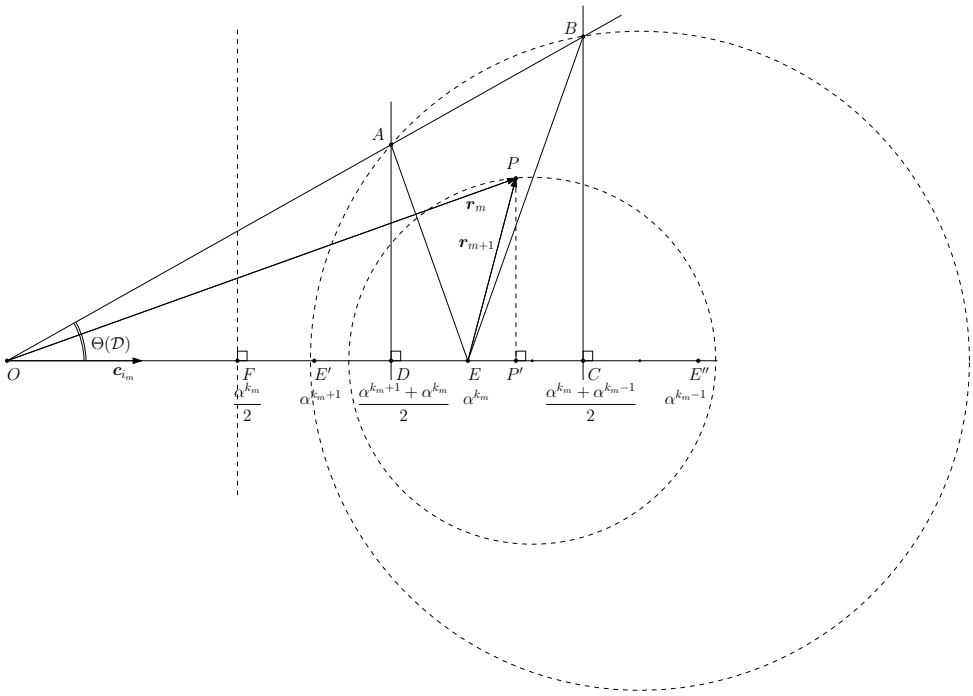

Fig. 15. Proof of Theorem 1

$$\bullet \text{ For point B: } \beta_B = \frac{\overline{BE}}{\overline{BO}} = \frac{\overline{AE}}{\overline{AO}} = \beta_A, \tag{81}$$

$$\text{since } \overline{BC} = \frac{\alpha^{k_{m-1}} + \alpha^{k_m}}{2} \tan\left(\Theta(\mathcal{D})\right) = \frac{\overline{AD}}{\alpha} \quad \text{(eq. (79))}$$

$$\overline{CE} = \frac{\alpha^{k_m-1} - \alpha^{k_m}}{2} = \frac{\overline{DE}}{\alpha} \quad \text{(eq. (80))}$$

$$\overline{BE}^2 = \overline{BC}^2 + \overline{CE}^2 = \frac{\overline{AD}^2 + \overline{DE}^2}{\alpha^2} = \frac{\overline{AE}^2}{\alpha^2}$$

$$\overline{BO} = \frac{\alpha^{k_m-1} + \alpha^{k_m}}{2} \sec\left(\Theta(\mathcal{D})\right) = \frac{\overline{AO}}{\alpha}.$$

$$\bullet \text{ For point C: } \beta_C = \frac{\overline{CE}}{\overline{CO}} = \frac{\frac{\alpha^{k_m-1}-\alpha^{k_m}}{2}}{\frac{\alpha^{k_m-1}+\alpha^{k_m}}{2}} = \frac{1-\alpha}{1+\alpha}. \tag{82}$$

$$\bullet \text{ For point D: } \beta_D = \frac{\overline{DE}}{\overline{DO}} = \frac{\frac{\alpha^{k_m}-\alpha^{k_m+1}}{2}}{\frac{\alpha^{k_m}+\alpha^{k_m+1}}{2}} = \frac{1-\alpha}{1+\alpha} = \beta_C. \tag{83}$$

Since $\alpha < 1$ and $\cos\left(\Theta(\mathcal{D})\right) \leq 1$, we have that

$$1 - \frac{4\alpha}{(1+\alpha)^2}\cos^2\left(\Theta(\mathcal{D})\right) \geq 1 - \frac{4\alpha}{(1+\alpha)^2} = \left(\frac{1-\alpha}{1+\alpha}\right)^2. \tag{84}$$

Then, from eqs. (78), (81), (82) and (83),

$$\beta = \max\{\beta_A, \beta_B, \beta_C, \beta_D\} = \sqrt{1 - \frac{4\alpha}{(1+\alpha)^2}\cos^2\left(\Theta(\mathcal{D})\right)}. \tag{85}$$

This completes the proof.

It is important to note that this proof can be trivially adapted to the case where $\|\boldsymbol{c}_{i_m}\| = \|\boldsymbol{c}\| \neq 1$. Referring to Fig. 15, we just need to change the common factor α^{k_m} in the lengths of $\overline{OF}$, $\overline{OE'}$, $\overline{OD}$, $\overline{OE}$, $\overline{OC}$ and $\overline{OE''}$ by $\|\boldsymbol{c}\|\alpha^{k_m}$. The rest of the proof, as well as eqs. (71) and (72), remain unchanged. ■

Eduardo A. B. da Silva (M'95, SM'05) was born in Rio de Janeiro, Brazil. He received the Electronics Engineering degree from Instituto Militar de Engenharia (IME), Brazil, in 1984, the M.Sc. degree in Electrical Engineering from Universidade Federal do Rio de Janeiro (COPPE/UFRJ) in 1990, and the Ph.D. degree in Electronics from the University of Essex, England, in 1995.

In 1987 and 1988 he was with the Department of Electrical Engineering at Instituto Militar de Engenharia, Rio de Janeiro, Brazil. Since 1989 he has been with the Department of Electronics Engineering (the undergraduate dept.), UFRJ. He has also been with the Department of Electrical Engineering (the graduate studies dept.), COPPE/UFRJ, since 1996. His teaching and research interests lie in the fields of digital signal, image and video processing. In these fields, he has published over 200 peer reviewed papers. He won the British Telecom Postgraduate Publication Prize in 1995, for his paper on aliasing cancellation in subband coding. He is also co-author of the book "Digital Signal Processing - System Analysis and Design", published by Cambridge University Press, in 2002, that has also been translated to the Portuguese and Chinese languages, whose second edition has been published in 2010.

He has served as associate editor of the IEEE Transactions on Circuits and Systems - Part I, in 2002, 2003, 2008 and 2009, of the IEEE Transactions on Circuits and Systems - Part II in 2006 and 2007, and of Multidimensional, Systems and Signal Processing, Springer since 2006. He has been a Distinguished Lecturer of the IEEE Circuits and Systems Society in 2003 and 2004. He was Technical Program Co-Chair of ISCAS2011. He is a senior member of the IEEE, of the Brazilian Telecommunication Society, and also a member of the Brazilian Society of Television Engineering. His research interests lie in the fields of signal and image processing, signal compression, digital TV and pattern recognition, together with its applications to telecommunications and the oil and gas industry.

Lisandro Lovisolo was born in Neuquen, Argentina, but considers himself brazilian. He received the Electronics Engineering degree from Universidade Federal do Rio de Janeiro, in 1999, the M.Sc. degree in Electrical Engineering in 2001, and the D.Sc degree in Electrical Engineering both from Universidade Federal do Rio de Janeiro (COPPE/UFRJ). Since 2003 he has been with the Department of Electronics and Communications Engineering (the undergraduate dept.), UERJ. He has also been with the graduate studies in Electronics Program, since 2008. His research interests lie in the fields of digital signal and image processing, communications, machine learning and computer science.

Alessandro J. S. Dutra (M?05) received the electronic engineering degree from the Instituto Militar de Engenharia (IME), Rio de Janeiro, Brazil, in 1993, the M.Sc. degree in electrical engineering from Universidade Federal do Rio de Janeiro (COPPE/UFRJ) in 1999, and the Ph.D. degree in electrical engineering from Rensselaer Polytechnic Institute (RPI), Troy, NY, in 2010.

He was a Post-Doctoral Researcher in the Department of Electrical Engineering, COPPE/UFRJ. He was part of the team that worked in the development of the Brazilian Digital Television System. Since 2012, he has been with GE Global Research, Brazil Technology Center, where he is a member of the Smart Systems research group. His research interests lie in the fields of data compression, digital signal and image processing, and machine learning for signal processing.

Paulo S. R. Diniz was born in Niterói, Brazil. He received the Electronics Eng. degree (Cum Laude) from the Federal University of Rio de Janeiro (UFRJ) in 1978, the M.Sc. degree from COPPE/UFRJ in 1981, and the Ph.D. from Concordia University, Montreal, P.Q., Canada, in 1984, all in electrical engineering.

Since 1979 he has been with the Department of Electronic Engineering (the undergraduate dept.) UFRJ. He has also been with the Program of Electrical Engineering (the graduate studies dept.), COPPE/UFRJ, since 1984, where he is presently a Professor. He served as Undergraduate Course Coordinator and as Chairman of the Graduate Department. He has received the Rio de Janeiro State Scientist award, from the Governor of Rio de Janeiro state.

From January 1991 to July 1992, he was a visiting Research Associate in the Department of Electrical and Computer Engineering of University of Victoria, Victoria, B.C., Canada. He also held a Docent position at Helsinki University of Technology. From January 2002 to June 2002, he was a Melchor Chair Professor in the Department of Electrical Engineering of University of Notre Dame, Notre Dame, IN, USA. His teaching and research interests are in analog and digital signal processing, adaptive signal processing, digital communications, wireless communications, multirate systems, stochastic processes, and electronic circuits. He has published several refereed papers in some of these areas and wrote the text books *ADAPTIVE FILTERING: Algorithms and Practical Implementation*, Fourth Edition, Springer, NY, 2013, and *DIGITAL SIGNAL PROCESSING: System Analysis and Design*, Second Edition, Cambridge University Press, Cambridge, UK, 2010 (with E. A. B. da Silva and S. L. Netto), and the monograph *BLOCK TRANSCEIVERS: OFDM and Beyond*, Morgan & Claypool, New York, NY, 2012 (W. A. Martins, and M. V. S. Lima).

He has served as the Technical Program Chair of the 1995 MWSCAS held in Rio de Janeiro, Brazil. He was the General co-Chair of the IEEE ISCAS2011, and Technical Program co-Chair of the IEEE SPAWC2008. He has also served Vice President for region 9 of the IEEE Circuits and Systems Society and as Chairman of the DSP technical committee of the same Society. He is also a Fellow of IEEE. He has served as associate editor for the following Journals: IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing from 1996 to 1999, IEEE Transactions on Signal Processing from 1999 to 2002, and the Circuits, Systems and Signal Processing Journal from 1998 to 2002. He was a distinguished lecturer of the IEEE Circuits and Systems Society for the year 2000 to 2001. In 2004 he served as distinguished lecturer of the IEEE Signal Processing Society and received the 2004 Education Award of the IEEE Circuits and Systems Society. He also holds some best-paper awards from conferences and from an IEEE journal. He has also received the 2014 Charles Desoer Technical Achievement Award from the IEEE Circuits and Systems Society.