Discrete Lagrangian Method for Optimizing the Design of Multiplierless QMF Filter Banks

Benjamin W. Wah, Yi Shang, and Zhe Wu *
Department of Electrical and Computer Engineering
and the Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA
{wah, shang, zhewu}@manip.crhc.uiuc.edu
URL: http://manip.crhc.uiuc.edu

Abstract

In this paper, we present a new discrete Lagrangian optimization method for designing multiplierless QMF (quadrature mirror filter) filter banks. In multiplierless QMF filter banks, filter coefficients are powers-of-two (PO2) where numbers are represented as sums or differences of powers of two (also called Canonical Signed Digit–CSD–representation), and multiplications can be carried out as additions, subtractions and shifting. We formulate the design problem as a nonlinear discrete constrained optimization problem, using the reconstruction error as the objective, and other performance metrics as constraints. One of the major advantages of this formulation is that it allows us to search for designs that improve over the best existing designs with respect to all performance metrics, rather than finding designs that trade one performance metric for another. We show that our design method can find designs that improve over Johnston's benchmark designs using a maximum of three to six ONE bits in each filter coefficient.

1: Introduction

Digital filter banks have been applied in many engineering fields. Their design objectives consist of their overall metrics and the metrics of each individual filter. Figure 1 summarizes the various design objectives for measuring design quality. In general, filter bank-design problems are multi-objective, continuous, nonlinear optimization problems.

Algorithms for designing filter banks are either optimization-based or non-optimization based. In optimization-based methods, a design problem is formulated as a multi-objective nonlinear optimization problem whose form may be application- and filter-dependent. The problem is then converted into a single-objective optimization problem and solved by existing optimization methods, such as gradient-descent, Lagrange-multiplier, quasi-Newton, simulated-annealing, and genetics-based methods. On the other hand, filter-bank design problems have been solved by non-optimization-based algorithms, which include spectral factorization and heuristic methods. These methods generally do not continue to find better designs once a suboptimal design has been found.

In this paper, we study global optimization methods to design multiplierless QMF filter banks. In a two-band QMF filter bank, the reconstructed signal is [3]:

$$\hat{X}(z) = \frac{1}{2}[H_0(z)F_0(z) + H_1(z)F_1(z)]\ X(z) + \frac{1}{2}[H_0(-z)F_0(z) + H_1(-z)F_1(z)]\ X(-z) \quad (1)$$

* Research supported by National Science Foundation Grant MIP 96-32316.
Proc. 1997 IEEE International Conference on Application-Specific Systems, Architectures and Processors.

Filter	Design Objectives
Overall Filter Bank	Minimize amplitude distortion
	Minimize aliasing distortion
	Minimize phase distortion
Single Filter	Minimize stopband ripple (δ_s)
	Minimize passpand ripple (δ_p)
	Minimize transition bandwidth (T_t)
	Minimize stopband energy (E_s)
	Maximize passband flatness (E_p)

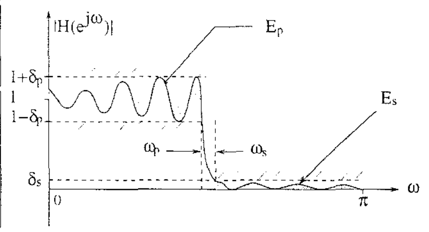

Figure 1. Possible design objectives of filter banks and an illustration of the design objectives of a single low-pass filter. ($[0, \omega_p]$ is the pass band; $[\omega_s, \pi]$, the stop band; $[\omega_p, \omega_s]$, the transition band.)

where $X(z)$ is the original signal, and $H_i(z)$ and $F_i(z)$ are, respectively, the response of the analysis and synthesis filters. To perfectly reconstruct the original signal based on $\hat{X}$, we have to eliminate aliasing, amplitude, and phase distortions. QMF FIR filter banks implement perfect reconstruction by setting $F_0(z) = H_1(-z)$, $F_1(z) = -H_0(-z)$ and $H_1(z) = H_0(-z)$, leading to a filter bank with one prototype filter $H_0(z)$, linear phase, and no aliasing distortions.

Traditional FIR filters in QMF filter banks use real numbers or fixed-point numbers as filter coefficients. Multiplications of such long floating point numbers generally limit the speed of FIR filtering. To overcome such a limitation, *multiplierless* (*powers-of-two* or *PO2*) filters have been proposed. These filters use filter coefficients that have only a few bits that are ones. When multiplying a filter input (multiplicand) with one such coefficient (multiplier), the product can be found by adding and shifting the multiplicand a number of times corresponding to the number of ONE bits in the multiplier. For example, multiplying x by 0100001001 can be written as the sum of three terms, $x \times 2^8 + x \times 2^3 + x \times 2^0$, each of which can be obtained by shifting x. A limited sequence of shifts and adds are usually much faster than full multiplications. Without using full multiplications, each filter tap takes less area to implement in VLSI, and more filter taps can be accommodated in a given area to implement filter banks of higher performance.

The frequency response of a PO2 filter, $H(z)$, is

$$H(z) = \sum_{i=0}^{m-1} x_i z^{-i} = \sum_{i=0}^{m-1} \left(\sum_{j=0}^{d-1} e_{i,j} 2^j \right) z^{-i} \quad \text{where} \sum_{j=0}^{d-1} |e_{i,j}| \leq l \text{ for all } i, \;\; e_{i,j} = -1, 0, 1. \tag{2}$$

Here, m is the length of the PO2 filter, l is the maximum number of ONE bits used in each coefficient, and d is the number of bits in each coefficient.

The design of multiplierless filters has been solved as integer programming problems which represent filter coefficients as variables with restricted values of powers-of-two. Other optimization techniques that have been applied include combinatorial search methods [10], simulated annealing [1], genetic algorithms [11], linear programming [6], and continuous Lagrange-multiplier methods in combination with a tree search [13].

In this paper, we present an efficient discrete Lagrange multiplier method (DLM) for designing multiplierless QMF filter banks. The paper is divided into six sections. In Section 2, we formulate the design problem as a single-objective constrained optimization problem. Section 3 summarizes the principle behind discrete Lagrangian optimization and present the discrete Saddle-Point Theorem. Section 4 examines issues related to the implementation of DLM to design multiplierless filter banks. Finally, Section 5 present experimental results.

2: Problem Formulation

The design of QMF filter banks can be formulated as a multi-objective unconstrained optimization problem or as a single-objective constrained optimization problem.

2.1: Multi-Objective Unconstrained Formulation

In a multi-objective formulation, the goals can be to

- Minimize the amplitude distortion (reconstruction error) of the overall filter bank, or
- Optimize the individual performance measures of the prototype filter $H_0(z)$.

One possible formulation using a subset of the measures in Figure 1 is as follows.

$$\text{Minimize} \quad E_r \quad \text{and} \quad E_s \tag{3}$$
$$\text{where} \quad E_r = \int_{\omega=0}^{\frac{\pi}{2}} \left(|H_0(e^{j\omega})|^2 + |H_0(e^{j(\omega-\pi)})|^2 - 1\right)^2 d\omega \quad \text{and} \quad E_s = \int_{\omega=\omega_s}^{\pi} |H_0(e^{j\omega})|^2 d\omega$$

Unfortunately, optimal solutions to (3) are not necessarily optimal solutions to the original problem.

In general, optimal solutions of a multi-objective problem form a *Pareto optimal frontier* such that one solution on this frontier is not dominated by another. One approach to find a point on the frontier is to optimize a weighted sum of all the objectives [5]. This approach has difficulty when frontier points of certain characteristics are desired, such as those with certain transition bandwidth. Different combinations of weights must be tested by trial and error until a desired solution is found. When the desired characteristics are difficult to satisfy, trial and error is not effective in finding feasible designs. In this case, constrained formulations should be used instead.

2.2: Single-Objective Constrained Formulation

In this formulation, constraints are defined with respect to a reference design. Constraint-based methods have been applied to design QMF filter banks in both the frequency [5] and the time domains [9]. In the frequency domain, the most often considered objectives are E_r, the reconstruction error, and δ_s, the stopband ripple. In the time domain, Nayebi [9] gave a time-domain formulation with constraints in the frequency domain and designed filter banks using an iterative time-domain design algorithm.

In this paper, we formulate the design of QMF filter banks in the most general form as a nonlinear constrained optimization problem:

$$\begin{aligned} &\text{Minimize } E_r \\ &\text{subject to } E_p \le \theta_{E_p} \qquad E_s \le \theta_{E_s} \qquad T_t \le \theta_{T_t} \\ &\qquad\qquad\quad \delta_p \le \theta_{\delta_p} \qquad \delta_s \le \theta_{\delta_s} \end{aligned} \tag{4}$$

where θ_{E_p}, θ_{E_s}, θ_{δ_p}, θ_{δ_s} and θ_{T_t} are constraint bounds found in the best known design (with possibly some bounds relaxed or tightened in order to obtain designs of different trade-offs). The goal here is to find designs whose performance measures are better than or equal to those of the reference design. Since the objective and constraints are nonlinear, the problem is multi-modal with many local minima.

The original optimization problem with inequality constraints (4) can be transformed into an optimization problem with equality constraints as follows:

$$\text{Minimize} \quad f(x) = V_{E_r} = \frac{E_r - \theta_{E_r}}{\theta_{E_r}} \tag{5}$$

$$\begin{aligned} \text{subject to} \quad V_{E_p} &= \max\left(\frac{E_p - \theta_{E_p}}{\theta_{E_p}}, 0\right) = 0 & V_{E_s} &= \max\left(\frac{E_s - \theta_{E_s}}{\theta_{E_s}}, 0\right) = 0 \\ V_{\delta_p} &= \max\left(\frac{\delta_p - \theta_{\delta_p}}{\theta_{\delta_p}}, 0\right) = 0 & V_{\delta_s} &= \max\left(\frac{\delta_s - \theta_{\delta_s}}{\theta_{\delta_s}}, 0\right) = 0 \\ V_{T_t} &= \max\left(\frac{T_t - \theta_{T_t}}{\theta_{T_t}}, 0\right) = 0 \end{aligned} \tag{6}$$

where x is a vector of discrete coefficients, θ_{E_r} is the reconstruction error of the best known design, and all the objective and constraints have been normalized with respect to the corresponding values of the best known design.

3: Lagrangian Methods

In this section we first summarize past work on Lagrangian methods for solving continuous constrained optimization problems. We then extend continuous Lagrangian methods to discrete constrained optimization problems [12].

3.1: Continuous Lagrangian Method

Lagrangian methods are classical methods for solving continuous constrained optimization problems [8]. We review briefly the theory of Lagrange multipliers.

Define an equality constrained optimization problem as follows:

$$\begin{array}{ll} \min_{x \in E^m} & f(x) \\ \text{subject to} & g(x) = 0 \end{array} \tag{7}$$

where $g(x) = (g_1(x), \cdots, g_n(x))$ are n constraints. Lagrangian function F is defined by

$$F(x, \lambda) = f(x) + \sum_{i=1}^{n} \lambda_i g_i(x) \tag{8}$$

where $\lambda = (\lambda_1, \cdots, \lambda_n) \in E^n$ are Lagrange multipliers.

A *saddle-point* (x^*, λ^*) of $F(x, \lambda)$ is defined as one that satisfies the following condition.

$$F(x^*, \lambda) \le F(x^*, \lambda^*) \le F(x, \lambda^*) \tag{9}$$

for all (x^*, λ) and all (x, λ^*) sufficiently close to (x^*, λ^*).

Saddle-Point Theorem. x^* is a local minimum to (7) if and only if there exists λ^* such that (x^*, λ^*) constitutes a saddle point of the associated Lagrangian function $F(x, \lambda)$.

Based on the Saddle-Point Theorem, numerical algorithms have been developed to look for saddle points corresponding to local minima in a search space. One typical method is to simultaneously do descents in the original variable space of x and ascents in the Lagrange-multiplier space of λ. The method can be written as a set of ordinary differential equations as follows:

$$\frac{dx}{dt} = -\nabla_x F(x, \lambda) \quad \text{and} \quad \frac{d\lambda}{dt} = \nabla_\lambda F(x, \lambda) \tag{10}$$

where t is an autonomous variable, and ∇ represents the gradient function.

3.2: Discrete Lagrangian Method (DLM)

Little work has been done in applying Lagrangian methods to solve discrete constrained combinatorial search problems [4]. The difficulty in traditional Lagrangian methods lies in the requirement of a differentiable continuous space to find a saddle point. In this section, we describe an extension of Lagrangian method in discrete space [12].

We first define the gradient in discrete space as follows:

$$\Delta_x F(x, \lambda) \;=\; \delta_x \tag{11}$$

where $F(x \ominus \delta_x, \lambda)^1$ has the minimum value among all $F(y, \lambda)$ for y sufficiently close to x.

In discrete space, the saddle point has the same definition as in (9) except that x^* and x must be discrete values. We also have the following saddle-point theorem for discrete space.

$^1\ominus$ is an operator for changing one point in discrete space into another with δ value. An example of $\ominus$ is the exclusive-OR operator.

```
1. Generate a starting point $x$
2. Set initial value of $\lambda$
3. while $x$ is not a saddle point or stopping condition has not been reached
4.     update $x$ to $x'$ only if this will result in $F(x', \lambda) < F(x, \lambda)$
5.     if condition for updating $\lambda$ is satisfied then
6.         $\lambda_i := \lambda_i + c \times g_i$, ($c > 0$ is real)
7.     end if
8. end while
```

Figure 2. $\mathcal{A}$: An implementation of DLM

Discrete Saddle-Point Theorem. x^* is a local minimum of a discrete constrained optimization problem if and only if there exists some λ^* such that (x^*, λ^*) constitutes a saddle point of the associated Lagrangian function $F(x, \lambda)$.

In order to show that the conversion of inequality constraints to equality constraints in (6) is correct, we must prove that the first order necessary conditions of the discrete Lagrangian function is satisfied when the solution x is at a local minimum of f subject to constraints $g(x) = 0$. Due to the way we have define discrete gradients, the first order necessary conditions can be proved to be satisfied. The proof is omitted due to space limitation.

Based on this theorem, we have developed the following method to find discrete saddle points.

Discrete Lagrangian Method (DLM) $\mathcal{A}$.

$$x^{k+1} = x^k \ominus \Delta_x F(x^k, \lambda^k) \tag{12}$$

$$\lambda^{k+1} = \lambda^k + c \times g(x^k) \tag{13}$$

where c is a positive real number.

It is easy to see that the necessary condition for $\mathcal{A}$ to converge is when $g(x) = 0$, implying that x is a feasible solution to the original problem.

4: Implementation Issues

Figure 2 shows our implementation of DLM for designing multiplierless filter banks. We explain each step in detail in this section.

4.1: Generating a Starting Point

There are two alternatives to select a starting point (Line 1 in Figure 2): choosing an existing PO2 QMF filter bank, or choosing a discrete approximation of an existing QMF filter bank with real coefficients. The first alterative is not always possible because not many such filter banks are available in the literature. In this section, we discuss the second alternative.

Given a real coefficient and b, the maximum number of ONE bits to represent the coefficient, we first apply Booth's algorithm to represent consecutive 1's using two ONE bits and then truncate the least significant bits of the coefficients. As an example, consider a binary fixed point number 0.10011101100. After applying Booth's algorithm and truncation, we can represent the number in 2 ONE bits:

$$0.10011101100 \Longrightarrow_{\text{Booth'sAlgorithm}} 0.1010001\bar{0}\bar{1}00 \Longrightarrow_{\text{Truncation}} 2^{-1} + 2^{-3}.$$

Previous work [7, 10, 2] shows that scaling has a significant impact on the coefficient-optimization process in PO2 filters. In our case, the performance of a PO2 filter obtained by truncating its real coefficients to a fixed maximum number of ONE bits is not as good as one whose real coefficients were first multiplied by a scaling factor. We illustrate this observation in the following example.

Table 1. Comparison of a PO2 filter bank obtained by truncating the real coefficients of Johnston's 32e QMF filter bank [5] to 3 bits and a similar PO2 filter bank whose coefficients were scaled by 0.5565 before truncation. (Performance has been normalized with respect to the performance of the original filter bank.)

Performance Metrics	E_r	E_p	E_s	δ_p	δ_s	T_t
Filter bank A with Truncated Coefficients	6.93	9.61	1.09	1.89	1.05	1.00
Filter bank B with Scaling and Truncation	0.99	1.08	0.96	1.20	0.98	0.99

```
1. LeastSum = $+\infty$
2. **for** ScaleFactor := 0.5000 **to** 1.0 **step** 0.0001
3.     Multiply each filter coefficient by ScaleFactor
4.     Get the PO2 form of the scaled coefficients
5.     Compute the weighted sum of constraint violation: sum := $\sum_{i=1}^{5} w_i \times g_i$
6.     **if** (sum < LeastSum) **then**
7.         LeastSum := sum
8.         BestScale := ScaleFactor
9.     **endif**
10. **endfor**
11. **return** BestScale
```

Figure 3. Algorithm for finding the best scaling factor, where w_i is the weight of constraint i.

Consider Johnston's 32e filter bank [5] as a starting point. Table 1 shows the metrics of two PO2 filters: Filter Bank A was obtained by truncating each of the original coefficients to a maximum of 3 ONE bits, whereas Filter Bank B was obtained by multiplying each of the coefficients by 0.5565 before truncation. Filter Bank B performs better and almost as good as the original design with real coefficients.

Figure 3 shows a simple but effective algorithm to find the proper scaling factor. We evaluate the quality of the resulting starting point by a weighted sum of its performance metrics. Since under most circumstances, the constraint on transition bandwidth is more difficult to satisfy, we give it a weight 100 and use 1 for the other four metrics.

Experimental results show that the algorithm in Figure 3 works well and fast, It is important to point out that scaling does not help when the number of ONE bits allowed to represent each coefficient is large. For instance, when the maximum number of ONE bits allowed is larger than 6, the performance of all the filters is nearly the same for all scaling factors.

4.2: Updating x

The value of x is updated in Line 4 of DLM in Figure 2. There are two ways in which x can be updated: greedy update and hill climbing. In greedy updates, the update of x leading to the maximum improvement of $F(x, \lambda)$ in (8) is found before an update is made. This approach is very time consuming and may not lead to the best filter bank when DLM stops. On the other hand, in hill climbing, x is updated as soon as an improvement in $F(x, \lambda)$ is found. This approach is efficient and generally leads to good designs. For this reason, we use hill climbing as our update strategy.

We process all the bits of all the coefficients in a round-robin manner. Suppose m is the filter length, l is maximum number of ONE bits that can be used for each coefficient, and the i'th coefficient is composed of l bits $b_{i,1}, b_{i,2}, \ldots, b_{i,l}$. We process the bits in the following order repetitively:

$$b_{1,1}, b_{1,2}, \ldots, b_{1,l}, b_{2,1}, \ldots, b_{m,1}, \ldots, b_{m,l}.$$

For each bit $b_{i,j}$, we perturb it to be a new bit $b'_{i,j}$ that differs from $b_{i,j}$ by either the sign or the exponent or both, with the condition that $b'_{i,j}$ is not the same in sign and exponent as another bit of the i'th coefficient. Using $b_{i,1}, \cdots, b_{i,j-1}, b'_{i,j}, \cdots, b_{i,m}$ while keeping other coefficients the same, we compute the new Lagrangian value $F(x', \lambda)$ using (8) and accept the change if $F(x', \lambda) < F(x, \lambda)$.

4.3: Initializing and Updating λ

The value of λ is initialized in Line 2 of DLM in Figure 2. To allow our experiments to be repeated and our results be reproduced easily, we always set λ to zero as our initial point.

Line 5 of DLM in Figure 2 is related to the condition when λ should be updated. In traditional Lagrangian methods on continuous variables, λ is updated in every iteration. This approach does not work in DLM because if λ were updated after each update of x, then the search behaves like random probing and restarts from a new starting point even before a local minimum is reached. For this reason, λ for violated constraints should be updated less frequently. In our implementation, we update λ every time three coefficients have been processed. Since λ is updated before all the filter coefficients have been perturbed, the guidance provided by λ may not be exact.

When updating λ before the search reaches a local minimum of $F(x, \lambda)$, we set c in Line 6 of DLM to be a normalized value as follows:

$$c = \frac{\theta_{speed}}{\max_{i=1}^{n} g_i} \tag{14}$$

where θ_{speed} is a real constant used to control the speed of increasing λ. Experimentally, we have determined θ_{speed} to be 0.6818.

When the search reaches a local minimum of $F(x, \lambda)$, perturbing all the bits in all the coefficients will result in no improvement of $F(x, \lambda)$. At this point, we need to update λ differently in order to bring the search out of the local minimum. This is done by choosing a proper value of c in Line 6 of DLM. If λ is increased too fast, then the search will restart from a random starting point. On the other hand, if the increase of λ is too small, then the trajectory remains in the current local minimum. Hence, we would like to set c so that it will bring the search out of the current local minimum. This means that, after λ has been changed to λ', there exists x' in the neighborhood of x such that

$$F(x, \lambda) \leq F(x', \lambda) \quad \text{and} \quad F(x', \lambda') < F(x, \lambda') \tag{15}$$

Replacing $F(x, \lambda)$ by $f(x) + \sum_{i=1}^{n} \lambda_i \times g_i(x)$ in (15), we get the condition before λ changes:

$$f(x) + \sum_{i=1}^{n} \lambda_i \times g_i(x) \leq f(x') + \sum_{i=1}^{n} \lambda_i \times g_i(x') \tag{16}$$

and that after λ_i is updated to $\lambda_i' = \lambda_i + c \times g_i(x)$:

$$f(x) + \sum_{i=1}^{n} (\lambda_i + c \times g_i(x)) \times g_i(x) > f(x') + \sum_{i=1}^{n} (\lambda_i + c \times g_i(x)) \times g_i(x') \tag{17}$$

where $g_i(x')$ is the new violation value of the i'th constraint at x'. After transformations, we get

$$0 \leq F(x', \lambda) - F(x, \lambda) < c \times \sum_{i=1}^{n} g_i(x) \times (g_i(x) - g_i(x')) \quad \text{or} \quad c > \frac{F(x', \lambda) - F(x, \lambda)}{\sum_{i=1}^{n} g_i(x) \times (g_i(x) - g_i(x'))} \tag{18}$$

As an example, consider in Figure 4a the violation of transition bandwidth T_t in a typical search based on constraints derived from Johnston's 32e filter bank [5]. Figure 4a shows that the value of the violation on T_t can be extremely small, in the order of 10^{-5} in the later part of the search. For such small violation values, the update of λ_{T_t} using c defined in (14) will result in a large number of iterations before the violation can be overcome. Using c defined in (18) to increase λ_{T_t}, we see in Figure 4b that λ_{T_t} jumps three times when the condition for updating λ was satisfied. These saved at least half of the total search time in order to find the solution.

Finally, we notice in Line 6 of DLM that λ is nondecreasing. This means that as the search progresses, the λ's grow, and more emphasis is placed on getting the search out of local minima. This approach fails when the λ's are so large that the search is brought to a totally new terrain

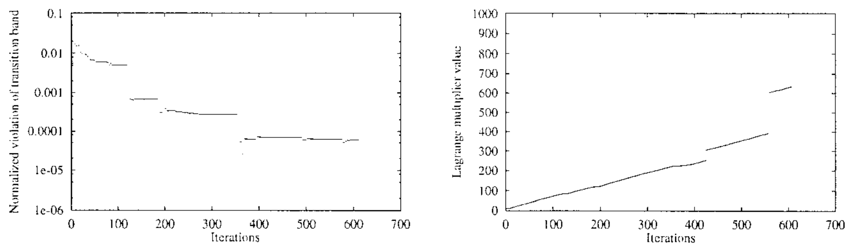

Figure 4. The violation values of T_t during a search (left) and the corresponding value of λ_{T_t} (right).

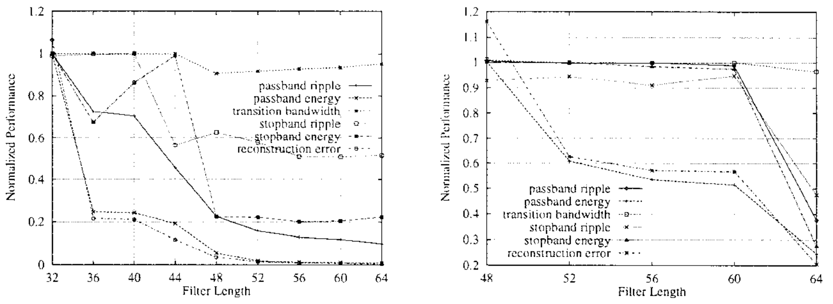

Figure 5. Normalized performance with respect to Johnston's 32e (left) and 48e (right) QMF filter banks [5] for PO2 filters with a maximum of 3 ONE bits per coefficient and different number of filter taps.

when Line 6 of DLM is applied. To cope with this problem, we measure the relative values between f and $\sum_{i=1}^{n} \lambda_i \times g_i$ and scale the λ's in order to keep the ratio within the following range.

$$0 \leq \frac{\sum_{i=1}^{n} \lambda_i \times g_i(x)}{f(x)} \leq \theta_{threshold} \tag{19}$$

If the ratio exceeds $\theta_{threshold}$, then we divide all the λ's by a constant r. In our experiments, we set $\theta_{threshold}$ to be 250 and r to be 1.3, and check $\theta_{threshold}$ every time when λ is updated.

5: Experimental Results

In this section, we compare the performance of PO2 QMF filter banks designed by DLM and those by Johnston [5], Chen *et al.* [2], Novel [14], simulated annealing, and genetic algorithms.

There are two parameters in a PO2 filter bank design: the maximum number of ONE bits in each filter coefficient and the number of filter taps. In our experiments, we have varied one while keeping the other fixed when evaluating a PO2 design with respect to a benchmark design.

We have used closed-form integration to compute the values of performance metrics. In contrast, Johnston [5] used sampling in computing energy values. Hence, designs found by Johnston are not necessarily at the local minima in a continuous sense.

We have evaluated PO2 designs obtained by DLM with respect to Johnston's designs whose coefficients are 32-bit real numbers. Using the performance of Johnston's 32e design as constraints [5], we ran DLM from 10 different starting points obtained by randomly perturbing 1% of all the coefficients of Johnston's design [5]. Each run was limited so that each ONE bit of the coefficient was processed in a round robin fashion 400 times. We then picked the best solution of the 10 runs and plotted the result in Figure 5, which shows the normalized performance of PO2 designs with increasing number of filter taps, while each filter coefficient has a maximum of 3 ONE bits. (The best design is one with the minimum reconstruction error if all the constraints are satisfied; otherwise, the one with the minimum violation is picked.) Our results show a design with 32 taps that is nearly as good as Johnston 32e's design. For filters with 32, 36, 40 and 44 taps, we used a

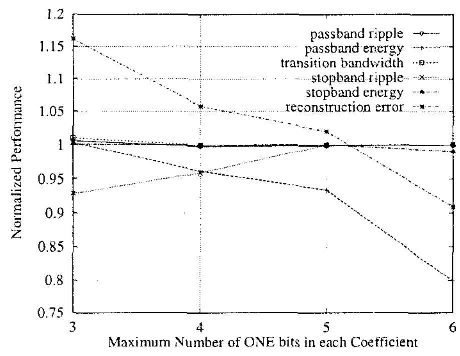

Figure 6. Normalized performance with respect to Johnston's 48e QMF filter bank [5] for PO2 filters with 48 taps and different maximum number of ONE bits per coefficient.

Table 2. Comparison of normalized performance of PO2 filter banks designed by DLM with respect to those designed by Johnston, Chen, Novel, simulated annealing (SA), and genetic algorithms (EA-Ct and EA-Wt). Columns 2-4 show the performance of DLM using 3 ONE bits for 32-tap filters and 6 ONE bits for 64-tap filters normalized with respect to that of Johnston's 32e, 64d, and 64e filter banks [5]. Columns 5-6 show the performance of DLM using 3 ONE bits normalized with respect to that of Chen *et al.*'s 64-tap and 80-tap filter banks [2]. Columns 7-10 show the performance of 32-tap filter banks designed using Novel [14], SA, and EA., normalized with respect to that of Johnston's 32e filter bank and using Johnston's design as constraints.

	Johnston's			Chen *et al.*		Other Search Methods wrt 32e			
Metrics	DLM–32e	DLM–64d	DLM–64e	DLM–64	DLM–80	Novel	SA	EA-Ct	EA-Wt
E_r	0.83	0.90	0.89	0.91	0.95	0.712	0.500	0.0943	0.507
E_p	1.00	0.82	0.83	0.80	0.96	0.896	0.582	1.542e5	0.590
E_s	1.00	1.00	1.00	1.00	0.86	1.000	1.000	1262	0.999
δ_p	1.00	0.97	1.00	1.00	1.00	1.000	1.000	1698	0.997
δ_s	0.99	0.75	1.00	1.00	1.00	1.000	1.000	17.30	0.999
T_t	1.00	1.00	1.00	1.00	1.00	1.000	1.013	0.000	1.013

starting point derived from Johnston's 32e design with filter coefficients first scaled by 0.5565 and truncated to a maximum of 3 ONE bits, and the filter coefficients of the remaining taps set to zeroes initially. Starting points for filters with longer than 44 taps were generated similarly, except that a scaling factor of 0.5584 was used instead. Our results show that, as the filter length is increased, all performance metrics improve, except the transition bandwidth, which remains close to that of the benchmark design.

With respect to Johnston's 48e design [5], we set a limit so that each ONE bit of the coefficient was processed in a round-robin fashion 800 times, and ran DLM once from the truncated Johnston's 48e design. (The scaling factor was 0.5584 for filters with 48, 52, 56, and 60 taps. The scaling factor was 0.6486 for the filter with 64 taps.) Our results show that our 48-tap PO2 design is slightly worse than that of Johnston's, while PO2 designs with 52 taps or larger have performance metrics that are either the same or better than those of Johnston's 48e design. In particular, the reconstruction error of our 52-tap PO2 design is 62% of Johnston's 48e design, while that of our 64-tap PO2 design is only 21% of Johnston's 48e design.

In the next set of experiments, we kept the same number of taps as Johnston's 48e design and increased the maximum number of ONE bits in each coefficient from 3 to 6. We set a limit so that each ONE bit of the coefficient was processed in a round-robin fashion 800 times, and ran DLM once from the truncated Johnston's 48e design. Figure 6 shows a design that is better than Johnston's 48e design when the maximum number of ONE bits per coefficient is 6. In this case, the reconstruction error is 91% of Johnston's 48e design. (The scaling factors used are 0.5584 for 3 bits, 0.8092 for 4 bits, 0.7409 for 5 bits, and 1.0 for 6 bits.)

With respect to Johnston's 64d and 64e designs, Table 2 shows improved PO2 designs obtained by DLM using a maximum of 6 ONE bits per coefficient and 64 taps. No improvements were found when the maximum number of ONE bits is less than 6.

Table 2 further shows improved PO2 designs obtained by DLM with respect to Chen *et al.*'s PO2 designs (with 3 ONE-bit coefficients) with 64 and 80 taps, respectively, and a maximum of 3 ONE bits per coefficient. In these designs, we used Chen *et al.*'s designs as starting points and ran DLM once with a limit so that each ONE bit was processed in a round-robin fashion 1,000 times.

Finally, we compare in Table 2 the performance of 32e PO2 filter banks obtained by DLM with a maximum of 3 ONE bits per coefficient, and those obtained by Novel, simulated annealing (SA), and evolutionary algorithms (EAs). Novel uses a continuous trace function to bring a search out of local minima rather than restarting the search from a new starting point when the search finds a feasible design. The SA we have used is SIMANN from netlib that works on a weighted-sum formulation. The EA is Sprave's Lice (Linear Cellular Evolution) that can be applied to both constrained and weighted-sum formulations. SA and EA-Wt use weighted-sum formulation with weight 1 for reconstruction error and weight 10 for the remaining metrics. EA-Ct works on the same constrained formulation as defined in (4). All methods were run significantly long with over 10 hours on a SUN SPARC20 workstation in each run.

We have tried various parameter settings and report the best solutions in Table 2. Novel improves Johnston's solutions consistently. SA results in larger transition bandwidths than Johnston's. EA-Wt has difficulty in improving over Johnston's design across all measures. EA-Ct converges to designs with very small reconstruction errors, while other constraints are violated significantly.

To summarize, we have presented a new discrete Lagrangian method for designing multiplierless powers-of-two (PO2) QMF filter banks. Our design method is unique because it starts from a constrained formulation with the objective of finding a design that improves over a benchmark design. Our design method is effective because it finds designs with very few ONE bits for each filter coefficient, allowing a cost-effective design to be implemented.

References

[1] R. A. Caruana and B. J. Coffey. Searching for optimal FIR multiplierless digital filters with simulated annealing. *Technical report, Philips Laboratories*, 1988.

[2] C.-K. Chen and J.-H. Lee. Design of linear-phase quadrature mirror filters with powers-of-two coefficients. *IEEE. Trans. Circuits Syst.*, 41(7):445–456, 7 1994.

[3] N. J. Fliege. *Multirate Digital Signal Processing.* John Wiley and Sons, 1994.

[4] A. M. Geoffrion. Lagrangian relaxation and its uses in integer programming. *Mathematical Programming Study*, 2:82–114, 1974.

[5] J. D. Johnston. A filter family designed for use in quadrature mirror filter banks. *Proc. of Int'l Conf. on ASSP*, pages 291–294, 1980.

[6] D. Kodek and K. Steiglitz. Comparison of optimal and local search methods for designing finite wordlength FIR digital filters. *IEEE Trans. Circuits Syst.*, 28:28–32, 1 1981.

[7] Y. C. Lim and S. R. Parker. FIR filter design over a discrete power-of-two coefficient space. *IEEE Trans. Acoust. Speech, Signal Processing*, ASSP-31:583–519, June 1983.

[8] D. G. Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley Publishing Company, 1984.

[9] K. Nayebi, T. P. Barnwell III, and M. J. T. Smith. Time-domain filter bank analysis: A new design theory. *IEEE Transactions on Signal Processing*, 40(6):1412–1429, June 1992.

[10] H. Samueli. An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients. *IEEE Transactions on Circuits and Systems*, 36(7):1044–1047, 1989.

[11] J. D. Schaffer and L. J. Eshelman. Designing multiplierless digital filters using genetic algorithms. In *Proc. Int'l Conf. on Genetic Algorithms*, pages 439–444, San Mateo, CA, 1993. Morgan Kaufmann.

[12] Y. Shang and B. W. Wah. A discrete lagrangian-based global-search method for solving satisfiability problems. *J. of Global Optimization*, (accepted to appear) 1997.

[13] J.-J. Shyu and Y.-C. Lin. A new approach to the design of discrete coefficient FIR digital filters. *IEEE Transactions on Signal Processing*, 41(1), 1 1995.

[14] B. W. Wah, Y. Shang, T. Wang, and T. Yu. Global optimization of QMF filter-bank design using NOVEL. In *Proc. Int'l Conf. on Acoustics, Speech and Signal Processing*, volume 3, pages 2081–2084. IEEE, April 1997.