DESIGN OF MULTI-CHANNEL LINEAR PHASE FIR FILTERS

GUSTAV LJUNGARS[*] AND MINYUE FU[†]

Abstract. This paper examines the use of the Remez Exchange Algorithm for Multi-channel FIR filter design.

AMS subject classification. 93A25.

1. Introduction. The design of one-channel linear-phase FIR digital filters was a hot topic in the early '70s. Especially the Chebyshev approximation attracted great interest, since it is optimal and the optimality is also an appealing one, because the optimization error is evenly distributed over frequency. [9] looked at linear programming techniques, and a few others proposed a whole range of different approaches to the problem. A major breakthrough was achieved in [7] and [6] by the introduction of the application of Remez exchange algorithm [1].

In 1995, an attempt was made to generalize the Parks-McClellan algorithm [4]. The proposed method extends the usage beyond the linear-phase region, enabling the user to specify filters with arbitrary magnitude *and* phase response. It has some limitations, however, the most important being that in the general case, the algorithm is optimal on a subset of the desired interval only.

Not much research has been performed on *global* optimization of filters, i.e. when optimality of an entire system, consisting not only of the filter, but also of other components, is opted for. In the original paper presenting the Parks-McClellan algorithm, only piecewise-linear specification functions is regarded. As is evident from reading their sources, this is actually an unnecessary requirement. In fact, any continuous requirement function is allowed, opening the possibility to optimize entire systems, where the actual filter forms one component only.

The developments in the field of telecommunications have caused multi-channel connections to become very common. Naturally, signals in such connections needs to be maintained and replenished regularly just like for one-channel systems. This calls for multi-channel filters and, indirectly, design algorithms for these. What makes these special is that channel interference has to be accounted for.

The global least-squares optimization problem for multi-channel filters is solved (see, e.g. [2]), so the least-squares norm will not be regarded in this report. For other matrix norms, there are only iterative and, due to the nature of the problem, very inefficient methods available. Encouraged by the unprecedented performance of the one-channel Remez exchange algorithm, we examine the multi-channel problem thoroughly with the hope to find a similarly efficient method.

1.1. Problem formulation. Figure 1.1 shows a one- or multi-channel system H consisting of a linear-phase *interference function* G and a *linear-phase filter* F (see Section 2.1 for a discussion on the properties of linear-phase systems). The frequency-domain response $Y(\omega)$ to a signal $X(\omega)$ is

$$Y(\omega) = FZ(\omega) = F(\omega)(GX(\omega)) = HX(\omega), \quad H(\omega) = FG(\omega).$$

[*] Department of Applied Electronics, Lund Institute of Technology, Lund University, Sweden

[†] Department of Electrical and Computer Engineering, The University of Newcastle, N.S.W. 2308 Australia

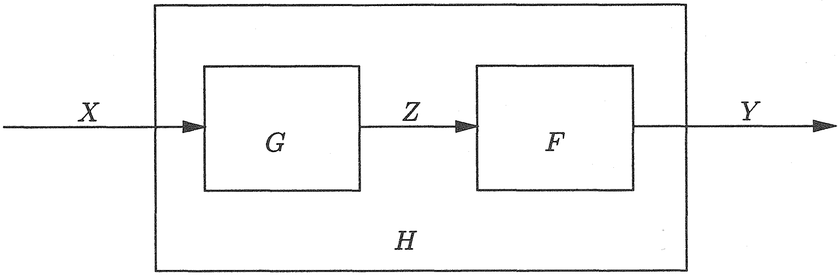

FIG. 1.1. *An interference function and a filter*

We want the overall magnitude response function $H(\omega) = FG(\omega)$ to be the best possible approximation to a specified, desired magnitude response function $H_{\mathrm{dr}}(\omega)$. This function is called the *specification function.* The problem can be recast into an optimization problem (α are the filter coefficients):

$$\text{minimize}_{\alpha} \|E(\omega)\| = \text{minimize}_{\alpha} \|FG(\omega) - H_{\mathrm{dr}}(\omega)\|, \quad \forall \omega \in [0, \omega]^1, \tag{1.1}$$

where $E(\omega)$ is the *error function*, and $\|\cdot\|$ denotes whatever norm we deem suitable. Please note that the design we are looking for is the best design for a system consisting *both* of an interference function *and* a filter, ie. we want to optimize the performance of *the entire system*, not just the filter.

The discussion will not be restricted to one-channel systems, but the systems are required to have an equal number of input and output channels.

EXAMPLE 1.1. *A telecommunications problem.* Imagine a long telecommunications link consisting of two coaxial cables lying next to each other in the ground. Each cable carries one phone call. It is reasonable to expect the signal to deteriorate by distance, so we had better put "repeaters", electronic devices designed to amplify and recondition the signal, along the way. The deterioration of the signal is modelled with the two-channel magnitude function $G(\omega)$ depicted in Figure 1.2. We have modelled signal deterioration within channels as being inductive (low-pass) in character whereas inter-channel distortion is more likely to be capacitive (high-pass) in character.

Figure 1.3 shows the desired magnitude response function *for the entire system* H_{dr}. Obviously, we want to attenuate external noise present in the system, a noise which is often of high pitch. This is accomplished by the low-pass character of the diagonal elements of the filter. Furthermore, we would rather see all interference between channels annihilated, which is reflected by the all-stop look of the off-diagonal elements.

Figure 1.4 shows one solution to the problem. It is the resulting filter of a maximum absolute value (MAV) norm optimization, carried out as a linear program (LP), see Section 4.2. The filter error function, the total system response, and the total system error function are shown in Figures 1.5, 1.6, and 1.7, respectively. The tap sizes for the filters are 39 taps for diagonal elements and 29 for off-diagonal elements.

1.2. Different Approaches Attempted. A number of different approaches to the problem have been explored. A wide range of methods are used hereby, from very

[1] So-called don't-care regions are never used in this report. For global optimization problems, a controlled behavior of the filter in the don't-care regions cannot be guaranteed.

general optimization methods with a solid theoretical background, to more specialized methods exploiting known properties of the problem—here, the theoretical foundation might sometimes be a bit thinner.

To summarize, this is the contents of the rest of the paper:

Preliminaries This section briefly introduces and discusses theory and notation needed and used throughout the rest of the report. Issues covered are linear-phase filters and the error function.

The One-Channel Filter Describes the Remez Exchange algorithm, the most popular method available for the design of Chebyshev-optimal one-channel linear-phase FIR filters, as well as the element-wise approach to the multi-channel problem.

Normed Optimization In all instances, the goal is to achieve optimum performance of the filter in respect to a (known or unknown) norm. However, this norm might not be present at all throughout the algorithm. This is very much the case for the Remez Exchange Algorithm. In this section we take a close look at algorithms where we *do* use a norm, ie. a matrix norm, to summarize all the complex behavior of a matrix transfer function into one single scalar error function, which is then optimized. We end up with either a linear program (LP) or a semidefinite program (SDP), which are both very well-known optimization problems with an extensive theoretical foundation and a multitude of efficient algorithms available.

The Multi-Channel Remez Exchange Algorithm No matter how efficient the algorithms for LPs and SDPs might be, they suffer from the same basic weakness: The error function has to be optimized over a very dense set of frequencies, which tends to make the problem formulation very large and also very quickly growing (sometimes exponentially) with filter length and other parameters involved. Therefore, an attempt has been made to reduce the number of frequencies to optimize for by choosing just a few which are expected to exhibit "extremal" behavior. The error function is optimized for these frequencies, and an exchange policy is also incorporated so those frequencies that cease to behave "extremally" can be discarded in favor for those who pop up during the course of the algorithm. This is very much the same approach as the Remez exchange algorithm in the one-channel case, except that the behavior of the algorithm has to be a bit more complex to accommodate for the interdependency between channels.

2. Preliminaries.

2.1. Linear-Phase Filters.

The z-transform of an M-tap FIR filter F is:

$$\mathcal{F}(z) = \sum_{t=0}^{M-1} h(t) z^{-t},$$ [2]

where $h(t)$ is the filter *impulse response*. The frequency response function is

$$\mathcal{F}(e^{j\omega}) = \sum_{t=0}^{M-1} h(t) e^{-j\omega t}.$$

[2] The letter t is chosen to denote "tap"—this should cause no confusion since no time domain discussions are carried out in the report.

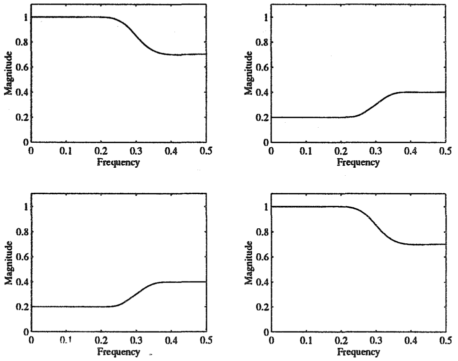

FIG. 1.2. *A two-channel interference function, $G(\omega)$*

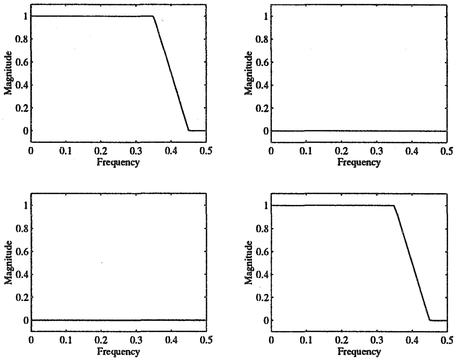

FIG. 1.3. *Specification, H_{dr}, for the two-dimensional telecommunications example*

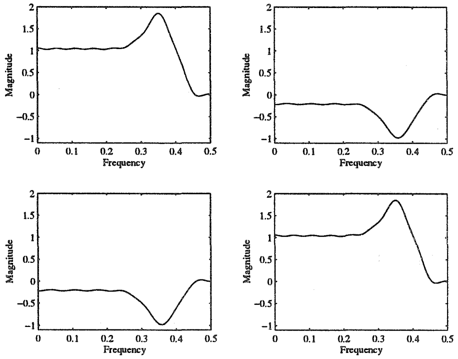

FIG. 1.4. *The real-valued frequency response function of a two-channel filter*

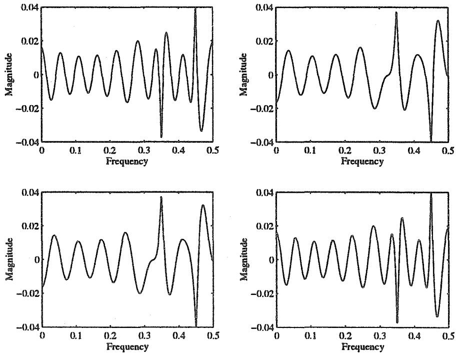

FIG. 1.5. *Filter error function*

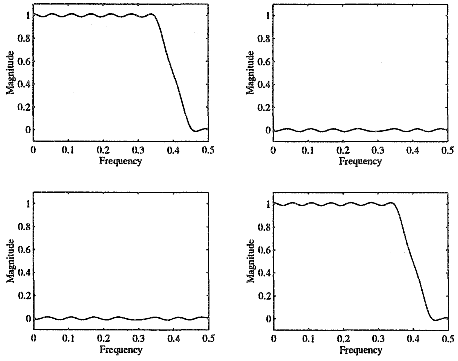

FIG. 1.6. *Total system response*

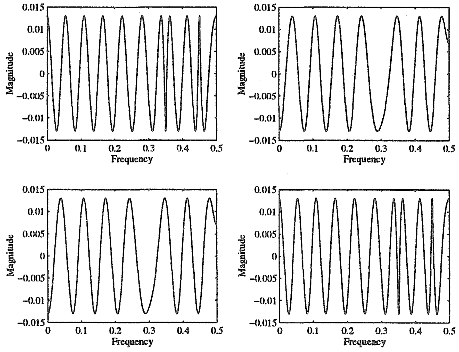

FIG. 1.7. *Total system error*

Assuming M odd, we can write this as

$$\mathcal{F}(e^{j\omega}) = e^{-j\omega\frac{M-1}{2}} \sum_{t=0}^{M-1} h(t)e^{-j\omega\left(t-\frac{M-1}{2}\right)}$$

$$= e^{-j\omega\frac{M-1}{2}}\left(h\left(\frac{M-1}{2}\right) + \sum_{t=0}^{\frac{M-3}{2}} h(t)e^{-j\omega\left(t-\frac{M-1}{2}\right)} + h(M-1-t)e^{j\omega\left(t-\frac{M-1}{2}\right)}\right).$$

It is easy to see that if we assume the symmetry condition $h(t) = h(M-1-t)$ to the impulse response, the above expression can be simplified further into

$$\mathcal{F}(e^{j\omega}) = e^{-j\omega\frac{M-1}{2}}\left(h\left(\frac{M-1}{2}\right) + \sum_{t=0}^{\frac{M-3}{2}} h(t)\cos\omega\left(t - \frac{M-1}{2}\right)\right)$$

$$= e^{-j\omega L}\sum_{t=0}^{L}\alpha(t)\cos\omega t = e^{-j\omega L}F(\omega), \tag{2.1}$$

$$F(\omega) = \sum_{t=0}^{L}\alpha(t)\cos\omega t, \quad L = \frac{M-1}{2} \tag{2.2}$$

where the α coefficients are:

$$\left\{\begin{array}{lcl} \alpha(0) & = & h(L) \\ \alpha(t) & = & 2h(L-t), \quad t = 1, 2, \cdots, L. \end{array}\right. \tag{2.3}$$

$F(\omega)$ is the *real-valued frequency response function.* The phase is a linear function of frequency, which means that the *group delay*[3] will be constant over frequency—no phase distortion is introduced by the filter, which is often an absolute requirement for signal processing applications. Equation 2.1 reassures us that the phase always will be linear, and in all further discussions, the phase will be disregarded, as will the complete frequency response function $\mathcal{F}(e^{j\omega})$ in favor of the simpler real-valued frequency response function $F(\omega)$.

With some simple modifications (see for instance [8, pp. 620–623] or [6]), Equations 2.1 and 2.3 can be generalized to apply to filters with an even number of taps M, and with an antisymmetric impulse response function ($h(t) = -h(M-1-t)$), as well.

2.2. Multi-Channel Linear-Phase Filters. The real-valued frequency response function for a linear-phase multi-channel filter is simply a matrix where each element is a real-valued frequency response function, so for a C-channel filter we get:

$$F(\omega) = \begin{pmatrix} F_{11}(\omega) & F_{12}(\omega) & \cdots & F_{1C}(\omega) \\ F_{21}(\omega) & F_{22}(\omega) & \cdots & F_{2C}(\omega) \\ \vdots & \vdots & \ddots & \vdots \\ F_{C1}(\omega) & F_{C2}(\omega) & \cdots & F_{CC}(\omega) \end{pmatrix} \tag{2.4}$$

where

$$F_{ij}(\omega) = \sum_{t=0}^{L_{ij}}\alpha_{ij}(t)\cos\omega t, \quad i, j = 1, 2, \cdots, C. \tag{2.5}$$

2.3. The Error Function. The error function is given by

$$E(\omega) = FG(\omega) - H_{\mathrm{dr}}(\omega). \tag{2.6}$$

[3] The group delay is the time it takes for a signal to pass through a filter.

Affine Format

The Semidefinite Programming (SDP) approach of Section 4.1 requires the error function to be in *affine* format, which is simply a linear combination of matrices. In this context, we choose to express the error primarily as a function of the filter coefficients, α, and we let the superscript $n \in 1, 2, \cdots, N$ denote for what frequency sample ω_n it is being evaluated:

$$\begin{aligned} E^{(n)}(\alpha) &= FG(\omega_n) - H_{\mathrm{dr}}(\omega_n) \\ &= -H_{\mathrm{dr}}(\omega_n) + \sum_{i=1}^{C}\sum_{j=1}^{C} \mathbf{K_{ij}G(\omega_n)} \sum_{\mathbf{t=0}}^{\mathbf{L_{ij}}} \alpha_{\mathbf{ij}} \cos\omega\mathbf{t}, \end{aligned} \tag{2.7}$$

where $\mathbf{K_{ij}}$ denotes a special matrix that has zeroes everywhere except for a one (1) in the (i, j) position.

Column Format

In Section 4.2 and Section 5 we will need Equation 2.6 expressed as a matrix times the filter coefficients α expressed as a vector:

$$A\mathrm{col}\,\alpha,$$

where

$$\mathrm{col}\,\alpha = \begin{pmatrix} \alpha_{11}(0) & \cdots & \alpha_{11}(L_{11}) & \alpha_{21}(0) & \cdots & \alpha_{21}(L_{21}) & \cdots & \alpha_{CC}(L_{CC}) \end{pmatrix}^T,$$

where col is the *column operator*, that converts a matrix to a single column vector simply by picking elements from the matrix column by column.

The Kronecker product can be used to implement the column operator, due to the following property:

$$\mathrm{col}\,FG(\omega) = (G(\omega)^T \otimes \mathbf{I_1})\mathrm{col}\,\mathbf{F}(\omega), \quad \mathbf{I_1} \in \mathbf{R^{m\times m}}$$

This is now applied to the transfer functions, which are all $C \times C$-matrix-valued (C is the number of channels) functions of frequency. Since col F is a vector of sums containing the α coefficients we wanted as a separate vector, we will have to continue further by using the Kronecker product once again, but we first define

$$\overline{\cos_L}\omega = \begin{pmatrix} 1 & \cos\omega & \cos 2\omega & \cdots & \cos L\omega \end{pmatrix},$$

so we get

$$\mathrm{col}\,F(\omega) = (\mathbf{I_2} \otimes \overline{\cos_{\mathbf{L}}}\omega)\mathrm{col}\,\alpha, \quad \mathbf{I_2} \in \mathbf{R^{C^2\times C^2}},$$

All in all we now yield

$$\mathrm{col}\,FG(\omega) = (G(\omega)^T \otimes \mathbf{I_1})(\mathbf{I_2} \otimes \overline{\cos_{\mathbf{L}}}\omega)\mathrm{col}\,\alpha = \hat{\mathbf{A}}(\omega)\mathrm{col}\,\alpha, \quad \mathbf{I_1} \in \mathbf{R^{C\times C}}, \tag{2.8}$$

where

$$\hat{A}(\omega) = (G(\omega)^T \otimes \mathbf{I_1})(\mathbf{I_2} \otimes \overline{\cos_{\mathbf{L}}}\omega), \quad \hat{\mathbf{A}}(\omega) \in \mathbf{R^{C^2\times C^2(L+1)}}. \tag{2.9}$$

3. The One-Channel Filter. The original paper [7] describing the application of the Remez exchange algorithm for the design of optimal linear-phase FIR filters refers to [1, pp. 72–100], for all the underlying theory. The bits of this that are of most importance for Section 5 further on will be summarized here. Please note that our treatise of the subject in general and the Remez exchange algorithm described in Section 3.2 below in particular is somewhat more general than the one of the original paper. The reason for this is that the authors of the original paper apparently did not foresee, or for that sake, cared about, the need of an algorithm that could find the optimal solution for an entire system—they were apparently only interested in finding the best possible filter for piecewise-constant specifications, and they could hereby make a few simplifying assumptions. These assumptions are however not very significant, as we will see, and we will consistently use and cite the more general theory from [1, pp. 72–100].

3.1. Introduction. Throughout this section, we will work with so-called generalized polynomials only:

$$H(\omega) = \sum_{t=0}^{L} \alpha(t)\beta_t(\omega), \tag{3.1}$$

where $\overline{\beta}$ is a vector of continuous basis functions β_t,

$$\overline{\beta} = \begin{pmatrix} \beta_0 & \beta_1 & \cdots & \beta_L \end{pmatrix}, \quad \beta_t \in \mathrm{C}[0,\pi], \quad t = 0, 1, \cdots, L, \tag{3.2}$$

that fulfill the Haar Condition.

DEFINITION 3.1. *The Haar Condition.* A vector of continuous basis functions β as described in Equation 3.2 is said to satisfy the Haar condition if a system of these function vectors, evaluated on any variable vector $\overline{\omega}$ of the same size,

$$\overline{\beta}(\overline{\omega}) = \begin{pmatrix} \beta_0(\omega_1) & \beta_1(\omega_1) & \cdots & \beta_L(\omega_1) \\ \beta_0(\omega_2) & \beta_1(\omega_2) & \cdots & \beta_L(\omega_2) \\ \vdots & \vdots & \ddots & \vdots \\ \beta_0(\omega_{L+1}) & \beta_1(\omega_{L+1}) & \cdots & \beta_L(\omega_{L+1}) \end{pmatrix}, \quad \overline{\omega} = \begin{pmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_{L+1} \end{pmatrix}, \tag{3.3}$$

is nonsingular. This is equivalent to saying that zero (0) is the only generalized polynomial (Eq. 3.1) that has $L+2$ or more roots on $[0,\pi]$.

Lemma 3.1. *For every* ordered *set of frequencies* $\overline{\omega}$, $0 \leq \omega_1 < \omega_2 < \cdots < \omega_{L+2} \leq \pi$ *the determinants* $\det \overline{\beta}(\overline{\omega})$ *all have the same sign.*

A natural choice of basis functions for signal processing applications would normally be trigonometric functions, eg.

$$\overline{\beta}(\omega) = \begin{pmatrix} 1 & \cos\omega & \cos 2\omega & \cdots & \cos L\omega \end{pmatrix},$$

but since we wish to optimize entire systems including interference functions, it is not feasible to restrict the discussion to that case.

We now want to find the generalized polynomial H that approximates a given function H_{dr} as well as possible in the Chebyshev sense, that is, we want to solve the following optimization problem:

$$\text{minimize}_{\alpha}||E|| \iff \text{minimize}_{\alpha}\left(\max_{\omega}\left|\sum_{t=0}^{L}\alpha(t)\beta_t(\omega) - H_{\mathrm{dr}}(\omega)\right|\right).$$

At the core of the solving of this problem efficiently is the alternation theorem below.

Theorem 3.1. The Alternation Theorem. *In order that a certain generalized polynomial*

$$H(\omega) = \sum_{t=0}^{L}\alpha_t\beta_t(\omega), \quad \omega \in [0,\pi]$$

shall be a best approximation to a given function $H_{\mathrm{dr}} \in \mathrm{C}[0,\pi]$ it is necessary and sufficient that the error function $E = H - H_{\mathrm{dr}}$ exhibit at least $L+2$ "alternations" thus:

$$E(\omega_n) = -E(\omega_{n-1}) = \pm\max_{\omega}|E(\omega)|, \quad \omega_1 < \omega_2 < \cdots < \omega_{L+2}. \tag{3.4}$$

Provided that the optimal frequency vector $\bar{\omega}$ is know, the optimal set of coefficients can be found simply by solving the full-rank linear equation system

$$\begin{aligned}
&E(\omega_n) = H(\omega_n) - H_{\mathrm{dr}}(\omega_n) = (-1)^n\lambda, \quad \forall n \in \{1,2,\cdots,L+2\}\\
\Leftrightarrow &\sum_{t=0}^{L}\alpha(t)\beta_t(\omega_n) - H_{\mathrm{dr}}(\omega_n) = (-1)^n\lambda, \quad \forall n \in \{1,2,\cdots,L+2\}\\
\Leftrightarrow &\begin{pmatrix} \beta_0(\omega_1) & \beta_1(\omega_1) & \cdots & \beta_L(\omega_1) & 1 \\ \beta_0(\omega_2) & \beta_1(\omega_2) & \cdots & \beta_L(\omega_2) & -1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \beta_0(\omega_{L+2}) & \beta_1(\omega_{L+2}) & \cdots & \beta_L\omega_{L+2} & (-1)^{L+2} \end{pmatrix}\begin{pmatrix} \alpha(0) \\ \vdots \\ \alpha(L) \\ \lambda \end{pmatrix} \qquad (3.5)\\
= &\begin{pmatrix} H_{\mathrm{dr}}(\omega_1) \\ H_{\mathrm{dr}}(\omega_2) \\ \vdots \\ H_{\mathrm{dr}}(\omega_{L+2}) \end{pmatrix}.
\end{aligned} \tag{3.6}$$

Note that λ is also an unknown in this expression, but since $\overline{\omega}$ was assumed to be the optimal frequency vector, $\delta = |\lambda| = ||E||$. Also note that even if $\overline{\omega}$ is *not* assumed to be the optimal one, the solution of the linear equation system of Equation 3.6 gives us the optimal solution for *that subset* of frequencies. Obviously, the trick is to iteratively find the subset on which subset optimality equals global optimality.

3.2. The Remez Exchange Algorithm. Below follows an outline of the famous Remez exchange algorithm. Figure 3.1 might help understanding the algorithm.

The Remez Exchange Algorithm

```
Input: The tap size $L \in \mathbf{R}$, the set of basis functions
    $\overline{\beta} \in \mathbf{C}^{L+1}[0, \pi]$, and an initial, ordered frequency set $\overline{\omega} \in \mathbf{R}^{L+2}$
    (this can be an arbitrary set, eg. a uniformly distributed one).
Output: The filter coefficients $\alpha \in \mathbf{R}^{L+1}$.
Minimize: Minimize the error function $E(\overline{\omega})$ by solving
    Equation 3.6. This yields a set of filter coefficients $\alpha'$ as well as
    the $\delta' = |\lambda'|$.
Evaluate: Evaluate the new error function $E'(\omega) = \sum_{t=0}^{L} \alpha' \beta_t(\omega)$.
    We will have
        $E'(\omega_n) = -E'(\omega_{n+1}) = \pm\delta', \quad n = 1, 2, \dots, L+1.$    (3.7)
Exchange:
    1. Due to (3.7) and continuity, $E'(\omega)$ has a root $z_n$ in each
       interval $[\omega_{n-1}, \omega_n], n = 2, 3, \cdots, L+2$. In addition, $z_1 = 0$,
       $z_{L+3} = \pi$. Let $\sigma_n = \operatorname{sign} E'(\omega_n)$.
    2. Select a trial set $\overline{\omega}'$ by finding local extrema of the error
       function on each subinterval defined by its roots:
           $\omega'_n = \max_{\omega \in [z_n, z_{n+1}]} \sigma_n E'(\omega).$
    3. While $||E'|| > \max_n |E'(\omega'_n)|$
       Do Define $\nu$ such that $E'(\nu) = ||E'||$. Insert $\nu$ in $\overline{\omega}'$ and
           remove a point such that the values of $E'(\omega'_n)$ still
           alternate in sign.
Repeat: Repeat from Minimize with the new set of extremal
    frequencies $\overline{\omega}'$ as long as the criterion of the alternation
    theorem (Eq. 3.4),
        $E(\omega_n) = -E(\omega_{n-1}) = \pm \max_{\omega} |E(\omega)|, \quad \omega_1 < \omega_2 < \cdots < \omega_{L+2},$
    is not satisfied.
```

The Exchange step can be implemented in several different ways. What is important is that the new $E'(\overline{\omega}')$ alternates in sign and that the largest peaks of the error function are included. A typical situation for the Exchange steps above is depicted in Figure 3.1

The convergence of the algorithm to the unique[4], optimal solution is governed by the following theorem.

Theorem 3.2. Convergence of the Remez Exchange Algorithm. *The successive generalized polynomials $H^{(k)}(\omega) = \sum_{t=0}^{L} \alpha^{(k)}(t)\beta_t(\omega)$ converge uniformly to the best approximation H^* according to the following inequality:*

$$||H^{(k)} - H^*|| \leq A\theta^k, \quad 0 < \theta < 1 \qquad (3.8)$$

[4] The uniqueness of the optimal solution is guaranteed by the Haar condition, see [1, pp. 80–82].

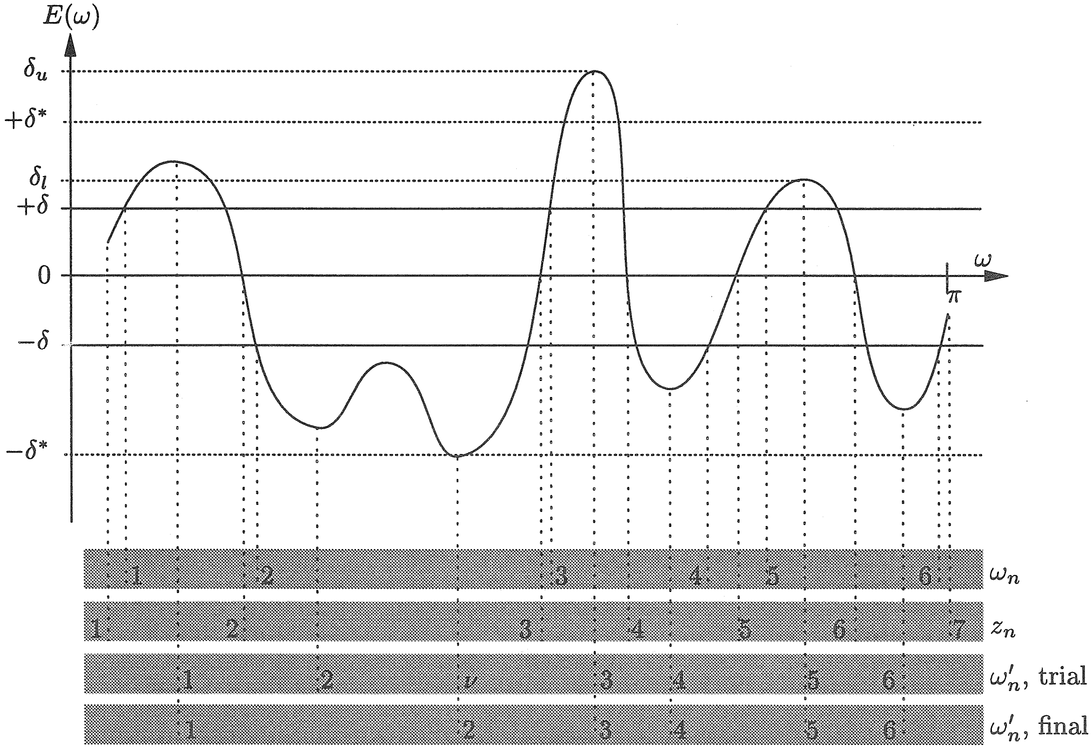

FIG. 3.1. *The error function, $E'(\omega) = \sum_{t=0}^{L} \alpha(t)\beta_t(\omega) - H_{dr}(\omega)$, in a step of the Remez exchange algorithm*

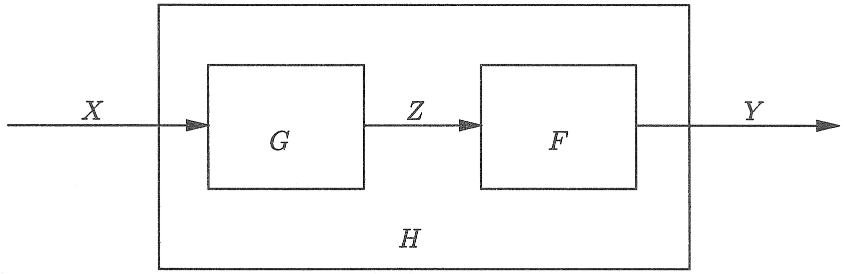

FIG. 3.2. *An interference function and a filter*

3.3. Application to One-Channel Filters. We like to solve the following optimization problem for the one-channel filter F including a continuous interference function G:

$$\text{minimize}_{\alpha}\|E\|, \iff \text{minimize}_{\alpha}\|FG - H_{\text{dr}}\|$$
$$\iff \text{minimize}_{\alpha}\left(\max_{\omega}\left|G(\omega)\sum_{t=0}^{L}\alpha(t)\cos\omega t - H_{\text{dr}}\right|\right). \qquad (3.9)$$

Obviously, the FG above is a generalized polynomial in the basis functions $\overline{\beta}$ defined as follows:

$$\overline{\beta}(\omega) = \left(\begin{array}{ccccc} G(\omega) & G(\omega)\cos\omega & G(\omega)\cos 2\omega & \cdots & G(\omega)\cos L\omega \end{array} \right). \tag{3.10}$$

By the Haar condition of Definition 3.1 it follows that G cannot have any zeros in the interval $[0, \pi]$.

Experiment.

EXAMPLE 3.1. *Element-wise Optimization of the Multi-Channel Filter.* Say for instance that we have the specification function $H_{\mathrm{dr}}(\omega)$ of Figure 3.3 and the interference function $G(\omega)$ depicted in Figure 3.4. If we use the Remez exchange algorithm individually for each element of the filter, we get the filter error function of Figure 3.5, which might look good enough. If however the effect of the interference function on the total system error is taken into account, things start to look much less encouraging, as in Figure 3.6.

Discussion.

The Remez exchange algorithm possesses a number of features that distinguishes it from other optimization algorithms and also contributes to its extreme efficiency:

- The **a priori knowledge** of the nature of the optimal solution provided by the alternation theorem (Th. 3.1) means that we know exactly what to look for. The alternation theorem tells us that as long as we can find the right extremal frequencies ω_n, we need to optimize the objective function for these frequencies only, and we need not bother about any of the other frequencies. This saves tremendous amounts of calculation and, consequently, time.
- The **convergence rate** given by Theorem 3.2 is very good, leading to, in most cases, less than 10 iterations.
- Each step of the algorithm involves **the solution of a full-rank linear equation system** and a **linear search** for local extrema of the error function. Both these steps can be very efficiently implemented.
- The algorithm is a **multiple exchange** algorithm, meaning that in the search for the "correct" extremal frequencies, all "trial" extremal frequencies are exchanged simultaneously at each step. This is much more efficient than exchanging only one frequency at a time.

4. Normed Optimization. From an optimization viewpoint, a *norm* can be defined as a way of summarizing the sometimes very complex behavior of a vector-valued or matrix-valued function into one single scalar function. Instead of trying to optimize all the elements and aspects of the original function, we can now concentrate on the simpler representation of it and hope that an optimal (in some sense) norm implies an optimal objective function (in some sense). Of course, what is meant by optimality is in practise defined by the norm used.

The choice of norm also determines the type and nature of the optimization problem we end up with. We have chosen to study these two different norms:

The Spectral Norm The maximum singular value (MSV) norm, or the *spectral*

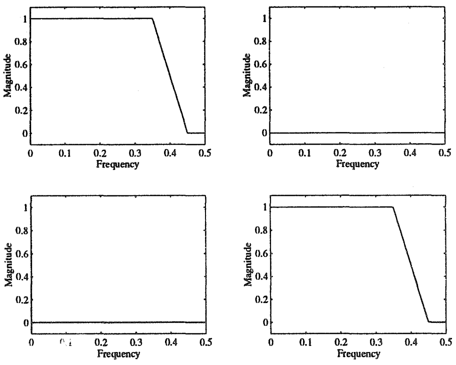

FIG. 3.3. *Specification function, $H_{dr}(\omega)$*

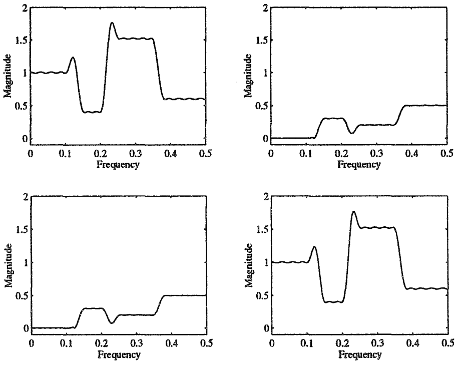

FIG. 3.4. *Interference function, $G(\omega)$*

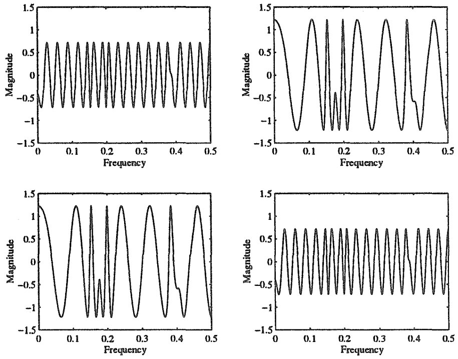

FIG. 3.5. *Filter error function*

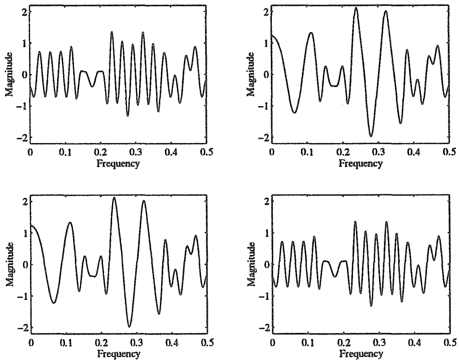

FIG. 3.6. *Total system error function, $E(\omega)$*

norm, is defined as

$$||A||_{MSV} = \max \sigma_i(A), \quad \forall i \in \{1, 2, \cdots, \min(m, n)\}. \tag{4.1}$$

where σ_i denote the singular values.

The optimization problem of minimizing the maximum singular value norm of a matrix-valued function can be recast as a semidefinite program (SDP), described in Section 4.1 below.

The Maximum Absolute Value (MAV) Norm For a matrix $A \in \mathbf{R}^{m \times n}$, the maximum absolute value (MAV) norm is defined as

$$||A||_{MAV} = \max |a_{ij}|, \quad \forall i \in \{1, 2, \cdots, m\}, \quad \forall j \in 1, 2, \cdots, n, \tag{4.2}$$

where a_{ij} denotes the (real-valued) elements of A.

The minimization of the MAV norm can be expressed as a linear program (LP), as explained in Section 4.2 later.

4.1. Semidefinite Programming (SDP). A semidefinite program (SDP) consists of a linear cost function which we wish to minimize and a linear matrix inequality (LMI) which expresses the constraints:

$$\begin{cases} \text{minimize}_x & c^T x \\ \text{subject to} & \hat{E}(x) \geq 0, \quad \hat{E}(x) \equiv \hat{E}_0 + \sum_{i=1}^{L} x_i \hat{E}_i, \end{cases} \tag{4.3}$$

where $x \in \mathbf{R}^L$ is the variable, and the $\{\hat{E}_i\}$ are a set of of square symmetric matrices of equal dimension. The expression for $\hat{E}(x)$ above is called an *affine* matrix expression. With $\hat{E}(x) \geq 0$ it is meant that the affine matrix $\hat{E}(x)$ has to be nonnegative definite, ie.

$$z^T \hat{E}(x) z \geq 0, \quad \forall z \in \mathbf{R}^m, \quad \hat{E}(x) \in \mathbf{R}^{m \times m}.$$

LMIs can be stacked diagonally. If, in the above example, we wish the variables to *simultaneously* meet

$$\hat{E}^{(n)}(x) \geq 0, \quad \forall n \in \{1, 2, \cdots, N\},$$

the constraint matrices can be stacked as one LMI $\bar{E}(x)$ like so:

$$\bar{E}(x) = \begin{pmatrix} \hat{E}^{(1)}(x) & 0 & \cdots & 0 \\ 0 & \hat{E}^{(2)}(x) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \hat{E}^{(N)}(x) \end{pmatrix}. \tag{4.4}$$

There are quite a few efficient algorithms for SDPs available. We have chosen to use one written by Stephen Boyd and Lieven Vandenberghe, see [10]. This is a package specifically written for SDPs in C, using optimized library routines[5] for the numerical linear algebra involved. The package is integrated with Matlab through the external interface, MEX.

To minimize the spectral (MSV) norm the original problem is recast into an SDP using Schur complements:

$$\text{minimize}_{\alpha} ||E^{(n)}(\alpha)||_{MSV}$$

can be written as

$$\left\{ \begin{array}{ll} \text{minimize}_{\alpha,\gamma} & \gamma \\ \text{subject to} & \begin{pmatrix} \gamma \mathbf{I} & E^{(n)}(\alpha) \\ E^{(n)}(\alpha)^T & \gamma \mathbf{I} \end{pmatrix} > 0 \end{array} \right.$$

$$\Longleftrightarrow \left\{ \begin{array}{ll} \text{minimize}_x & q^T x, \; q = \begin{pmatrix} 1 & 0 & \cdots & 0 \end{pmatrix}^T, \; x = \begin{pmatrix} \gamma & \alpha \end{pmatrix}^T \\ \text{subject to} & \hat{E}^{(n)}(x) > 0, \; \hat{E}^{(n)}(x) = \begin{pmatrix} \gamma \mathbf{I} & E^{(n)}(\alpha) \\ E^{(n)}(\alpha)^T & \gamma \mathbf{I} \end{pmatrix}. \end{array} \right. \tag{4.5}$$

Compared to Equation 1.1 we have changed the notation slightly. Where the error function normally can be regarded as a function of frequency, $E(\omega)$, it is here seen as a function of the filter coefficients, $E(\alpha)$, and since a discretization over frequencies will be needed sooner or later, it has been done now, and therefore $E^{(n)}(\alpha)$ denotes the error function evaluated at the frequency point ω_n. Furthermore, the affine form of E is used, see Subsection 2.3.

The spectral norm has to be minimized simultaneously for all frequencies, so the $\hat{E}^{(n)}(x), \;\; \forall n \in \{1, 2, \cdots, N\}$ are therefore stacked as in Equation 4.4 above to yield

[5] Specifically Netlib's BLAS and LAPACK, which for DEC AlphaStations are available as an optimized, and parallelized if necessary, library, `lmdx`.

the final optimization problem:

$$\begin{cases} \text{minimize}_x & q^T x \\ \text{subject to} & \hat{E}^{(n)}(x) > 0, \quad \forall n \in \{1, 2, \cdots, N\} \end{cases}$$

$$\iff \begin{cases} \text{minimize}_x & q^T x \\ \text{subject to} & \bar{E}(x) > 0, \quad \bar{E}(x) \text{ as in Eq. 4.4} \end{cases} \tag{4.6}$$

When the spectral norm is applied for all frequencies like in equation 4.6, it is commonly called the H_∞-norm.

Experiment.
A series of examples were ran using this method. The results are very consistent and the algorithm used is efficiently coded and seems to work very well for all problems attempted. Therefore we present only a typical result here.

We re-use the specification and interference functions of Example 3.1, Figures 3.3 and 3.4. The example 2-channel filter has 69 taps in its diagonal elements, and 29 taps in its off-diagonal elements. Using the algorithms of [10], the optimization took 325s to run on a DEC AlphaStation. The resulting plot can be seen in Figures 4.1–4.4.

Discussion.
As previously mentioned, since the performance of the filter has to be optimized over a dense set of frequencies, we tend to end up with very large matrices in the affine expression. Even if they are also very sparse, it is still a big problem that takes a long time to run even on very powerful workstations.

A few attempts were made with exchange algorithms. Instead of minimizing the error function norm for all frequencies, a few are chosen over which the error function norm is optimized. A new set of frequencies, which are the frequencies for which the error function norm is maximal, is selected and a new cycle is employed. The method is fairly hard to implement, since it is not known how many extrema to look for in the error function norm. If a fairly large number of extrema are chosen, the algorithm can still start to oscillate between to different sets of extremal frequencies. There are a number of more systematic methods available, such as [5] and [3], but the method still has to be dropped due to its notoriously bad performance, in particular when the number of channels grow large.

One might also question the optimality criterion. A quick look at Figure 4.3 reveals that the error is not very well distributed over frequencies, and even if the solution is clearly optimal in the H_∞ sense, it remains unclear whether this is actually the kind of optimality we want.

Finally, we point out a SDP approach to minimization of the spectral norm without the discretization of frequencies. This approach involves the use of a well-known bounded real lemma in the systems theory to replace ω with a positive-definite matrix P. The subsequent problem is a finite dimensional SDP problem and a numerical solution of polynomial complexity exists. The details of this approach can be found in [2]. However, this approach is not applied in our study because the dimension of P, is typically too large, rendering the numerical solution infeasible. This approach seems to be appliable only to cases where the tap sizes are quite small.

4.2. Linear Programming (LP). A linear program (LP), which is a special case of the semidefinite program described above, consists of a linear cost function

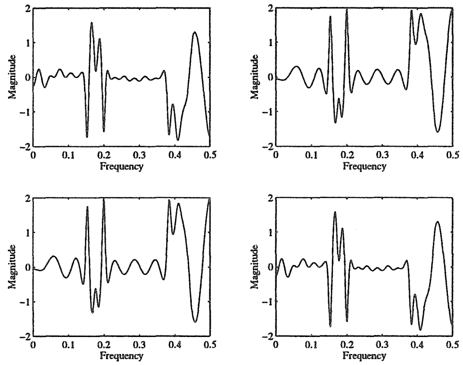

FIG. 4.1. *Filter error function*

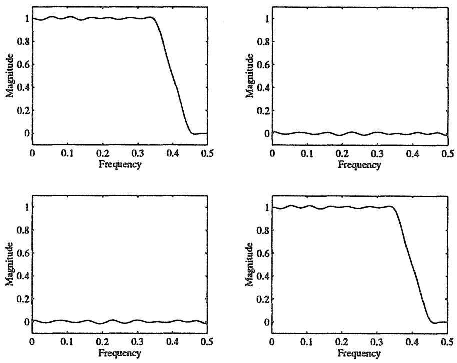

FIG. 4.2. *Total system response*

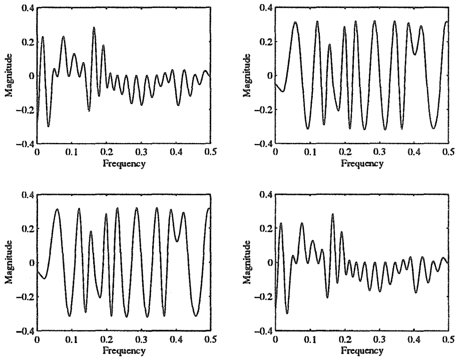

FIG. 4.3. *Total system error*

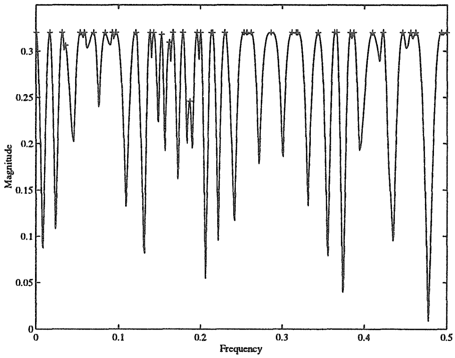

FIG. 4.4. *System error spectral norm*

and a set of linear constraints and looks like this in its *standard form*:

$$\left\{\begin{array}{ll} \text{minimize}_x & q^T x \\ \text{subject to} & Ax = b, \quad x \geq 0 \end{array}\right., \quad \left\{\begin{array}{ccl} q & \in & \mathbf{R}^{1\times n} \\ x & \in & \mathbf{R}^n \\ A & \in & \mathbf{R}^{m\times n} \\ b & \in & \mathbf{R}^m \end{array}\right. \tag{4.7}$$

The linear programming approach can be utilized to minimize the Chebyshev norm of a vector. If we for instance have an over-determined system of linear equations,

$$A\alpha = d, \quad \left\{\begin{array}{ccl} A & \in & \mathbf{R}^{m\times n} \\ \alpha & \in & \mathbf{R}^n \\ d & \in & \mathbf{R}^m \end{array}\right., \quad m > n,$$

it can be approximated in the Chebyshev sense by solving the optimization problem

$$\left\{\begin{array}{ll} \text{minimize}_{\alpha,\gamma} & \gamma \\ \text{subject to} & \left\{\begin{array}{ccccc} A\alpha & - & d & \leq & \mathbf{1}\gamma \\ A\alpha & - & d & \geq & -\mathbf{1}\gamma \end{array}\right., \quad \gamma \geq 0 \end{array}\right.$$

$$\Longleftrightarrow \left\{\begin{array}{ll} \text{minimize}_{\alpha,\gamma} & \gamma \\ \text{subject to} & \begin{pmatrix} A & -\mathbf{1} \\ -A & -\mathbf{1} \end{pmatrix} \begin{pmatrix} \alpha \\ \gamma \end{pmatrix} \leq \begin{pmatrix} d \\ -d \end{pmatrix}, \quad \gamma \geq 0 \end{array}\right. \tag{4.8}$$

In the above expressions, **1** is used to denote a column vector consisting of ones only.

When applied to multi-channel filter design, we want to minimize the MAV norm over all elements and all frequencies:

$$\text{minimize}_{\alpha} \left(\max_{n} \left(\max_{i,j} \left| FG_{ij}(\omega_n) - H_{dr_{ij}}(\omega_n) \right| \right) \right),$$

where $\omega_n, n = 1, 2, \cdots, N$ is a "dense" frequency set. Obviously, this problem can be restated as finding

$$\arg\min_{\alpha_+, \alpha_-, s, \gamma} \gamma \tag{4.9}$$

subject to

$$\begin{pmatrix} A & -A & -\mathbf{1} & \mathbf{I} \\ -A & A & -\mathbf{1} & \end{pmatrix} \begin{pmatrix} \alpha_+ \\ \alpha_- \\ \gamma \\ s \end{pmatrix} = \begin{pmatrix} d \\ -d \end{pmatrix}, \quad \left\{ \begin{array}{lcl} \gamma & \geq & 0 \\ \alpha_+ & \geq & 0 \\ \alpha_- & \geq & 0 \\ s & \geq & 0 \end{array} \right. ,$$

where A is related to Equation 2.9, d denotes the specification function H_{dr} evaluated for all ω_ns and written in column format, and $\operatorname{col} \alpha = \alpha_+ - \alpha_-$.

Note that we are effectively minimizing a Chebyshev norm, since all matrices are now written in column format. The MAV norm is hereby minimized indirectly.

Experiment.

A fair few simulation were carried through using the linear programming approach. The SDP solver package was used for the LPs as well due to its ease of use and stable behavior. The downside is however performance—you would expect dedicated LP solvers to run faster.

In figures 4.5–4.8, plots of a test run using the specification and interference functions of Example 3.1, Figures 3.3 and 3.4, are found. The filter has 69 taps in its diagonal elements and 29 taps in its off-diagonal elements. Using the LP algorithms of [10], the optimization took 708s to run on a DEC AlphaStation.

Discussion.

As for the SDP approach, since the performance of the filter has to be optimized over a dense set of frequencies, we tend to end up with very large matrices. The sparsity for the LP matrices are only around 10%, but they are on the other hand much smaller that the affine SDP matrices, and the number of nonzero elements are roughly the same.

Have a closer look at Figure 4.7. Evidently, all the error functions alternates between the same extremal values. The number of alternations differ between the channels, so clearly, this approach distribute the approximation error in a very uniform way.

5. The Multi-Channel Remez Exchange Algorithm. In this section we explore the possibilities of applying the general idea of the Remez exchange algorithm (as described in the beginning of Section 3.2) more or less unmodified to the multi-channel problem.

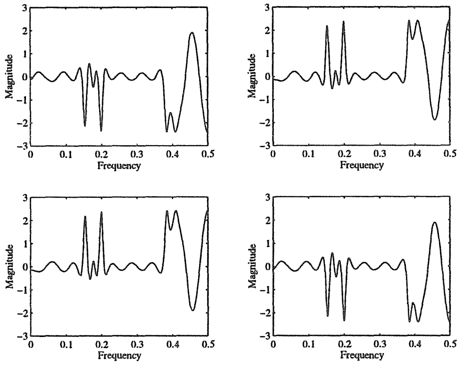

FIG. 4.5. *Filter error function*

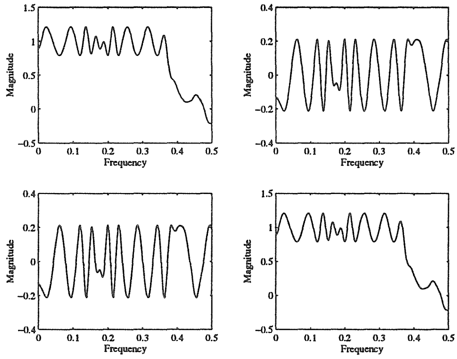

FIG. 4.6. *Total system response*

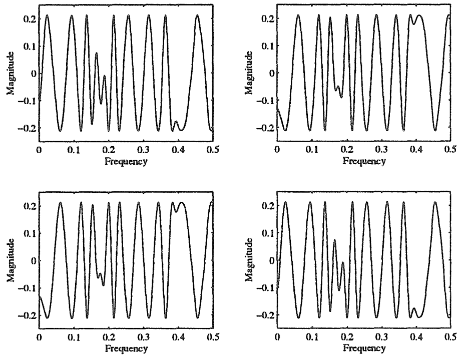

FIG. 4.7. *Total system error*

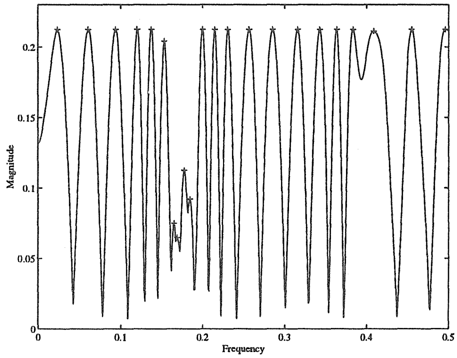

FIG. 4.8. *System error MAV norm*

It has to be emphasized that a new, working algorithm is *not* presented, but a few very promising simulations were carried through nevertheless.

The purpose of this section is to present an idea, or a concept, that, in the opinion of the authors, would be worth exploring further. The general approach is to boldly assume that the same theory holds for multi-channel filters as do for one-channel ones. Therefore, we assume that there exists a solution, in some aspect optimal, for which all the matrix elements of the error function of the final multi-channel filter alternates. However, the necessary theoretical extensions required to support the above assumptions are never made.

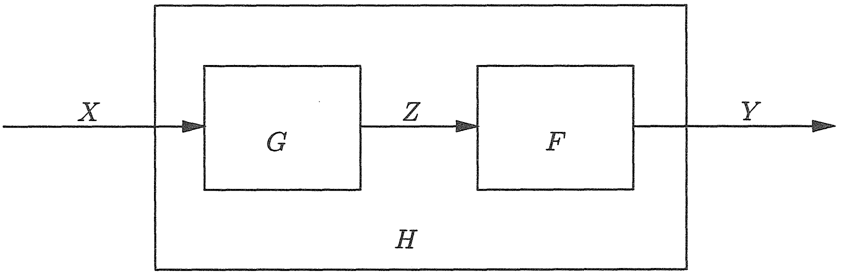

FIG. 5.1. *An interference function and a filter*

To be precise, We "enforce"

$$FG_{ij}(\omega_{ij_n}) - H_{dr_{ij}}(\omega_{ij_n}) = (-1)^n \lambda_{ij}, \quad n = 1, 2, \ldots, L_{ij} + 2 \tag{5.1}$$

simultaneously for all matrix elements (i, j). Please note that there are different ω_ns for different channels. For each channel (i, j), we would expect to find $L_{ij} + 2$ alternations in the error function. There are also different λs, one for each element. So, if Equation 5.1 holds, the system is optimal only in the sense that each element in the matrix transfer function $H(\omega)$ alternates between its extremum values. What sort of optimality in a strict mathematical sense this might equal to is not further investigated. Also note that the extremum values will be different for the different elements.

What distinguishes this problem from the one-channel problem is the fact that the matrix-valued interference function $G(\omega)$ introduces inter-channel interference, ie. the problem cannot be solved simply by solving an equation system like Equation 3.6 for each element, but has to be treated "holistically", simultaneously, treating all elements together.

The algorithm used forms one single equation system from the condition in Equation 5.1 by using the column operator as explained in Section 2.3. The resulting system is of the form

$$\begin{pmatrix} A & \Lambda \end{pmatrix} \begin{pmatrix} \alpha \\ \lambda \end{pmatrix} = b, \tag{5.2}$$

where λ are the λs from Eq. 5.1 above, written as a column vector, Λ is a matrix corresponding to the $(-1)^n$ factor from the same equation, A is the big matrix related to Equation 2.9, α are all the filter coefficients from all the filter matrix elements, written as one single vector, and b is H_{dr} for all frequencies and all matrix elements, written as one single vector as well.

Just like in the one-channel case, the system is solved (it is full-rank as long as $G(\omega)$ is nonsingular and the frequency vectors ω_{ij} contain no repeated values (Eq. 3.3)), after which the new error function is evaluated. New "extremal" frequencies are found, individually for each matrix element, in the same fashion as for one-channel filters (Section 3.2), and a new A matrix is formed. The above procedure is repeated until the "extremal" frequencies do not change any longer.

Constant Interference Function.

An interesting special case is when G is constant, ie. does not vary with frequency. In this particular case, the problem can be decoupled. To show this, we write Equation 5.1 in its full matrix form (please note that λ is a matrix):

$$F(\omega_n)G - H_{dr}(\omega_n) = (-1)^n\lambda \iff F(\omega_n) - H_{dr}(\omega_n)G^{-1} = (-1)^n\lambda G^{-1} \tag{5.3}$$

Clearly, since now $F(\omega_n)$ stands by itself, the problem is decoupled and can be solved as an independent problem for each matrix element (Chapter 3.1 & 3.2).

Experiment.

EXAMPLE 5.1. *A big multi-channel Remez algorithm example.* This is a really huge example. The filter has 69 taps in its diagonal and 29 in its off-diagonal elements and ten channels. The resulting equation system to be solved by the script has 1800 equations and variables, which means that the size of the system matrix exceeds three million elements! The matrix is sparse, but only to a limited extent—roughly 10%. In this context it has to be emphasized that the problem described in practise is far too big for any other method (as included in this report) to handle. The proposed algorithm however converged to an "optimal" (with the limitations as discussed above) solution in eight iterations and less than two minutes only! The resulting overall system magnitude response and error functions can be seen in Figures 5.2 and 5.3, respectively. Only a diagonal element and an off-diagonal element have been displayed in order to save space.

Discussion.

The algorithm proposed above works well for many simulated examples, especially when the cross-channel interference is not severe. However, reality is a bit cruel. For a fair few simulations with different tap lengths and different interference and requirement functions, the algorithm simply did not converge. Simple observation of the simulation script in verbose mode (which means that the error function is plotted for each step in the algorithm) suggests that the reason might be inter-channel interference, in the sense that the elements converge with different speeds and therefore, a slowly-converging element might disrupt the convergence of a faster-converging one, eventually causing an oscillation where a large error propagates back and forth between elements. A very interesting project would be to have a look at different techniques to control this.

6. Conclusions. Encouraged by the excellent performance of the Remez Exchange Algorithm when optimizing global performance of systems via Linear-Phase FIR Digital Filters in the one-channel domain, the possibilities to increase the performance accordingly for Multi-channel global optimization of systems were explored. Led by the results from this quest, a closer look was taken on the one-channel Remez algorithm. A few points are worth making:

- The problem already have an analytical solution for the least-squares norm [2]. Due to the down-sides of that norm (the Gibbs effect etc.), other norms are still of significant interest, even if iterative methods have to be applied for these.
- The general optimization algorithms available for solving linear and semidefinite programs are not efficient for solving these problems. Generally, one can observe that only a small fraction of all the frequencies that is optimized for, are required, had the "correct", ie. extremal frequencies been selected and non-extremal frequencies discarded in the optimization process.
- A very efficient algorithm for the multi-channel domain was proposed, but it does not work all the time. Further investigation into this algorithm is required.

REFERENCES

[1] E. W. Cheney, *Introduction to Approximation Theory*, Chelsea Publishing Company, New York, NY., second ed., 1982.

[2] M. Fu, S. Dasgupta, and G. A. Williamson, Algorithms for optimal multirate filter bank design, *Proceedings for the 4th international symposium on signal processing and its applications*, Aug. 1996, pp. 399–402.

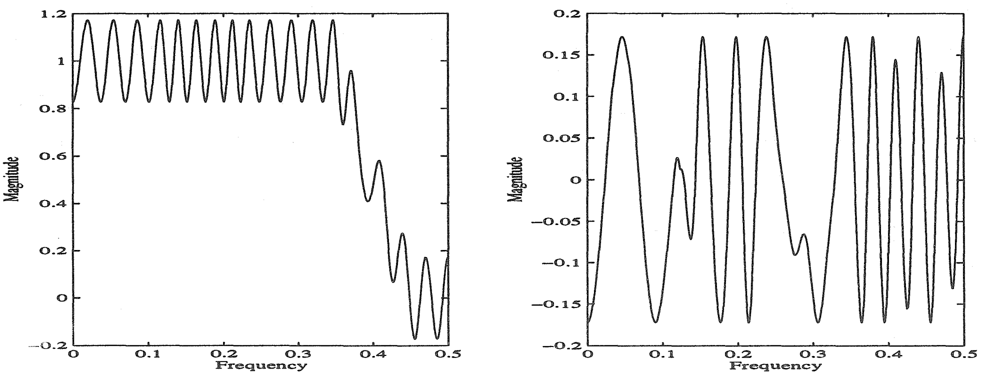

FIG. 5.2. *System response for diagonal and off-diagonal elements*

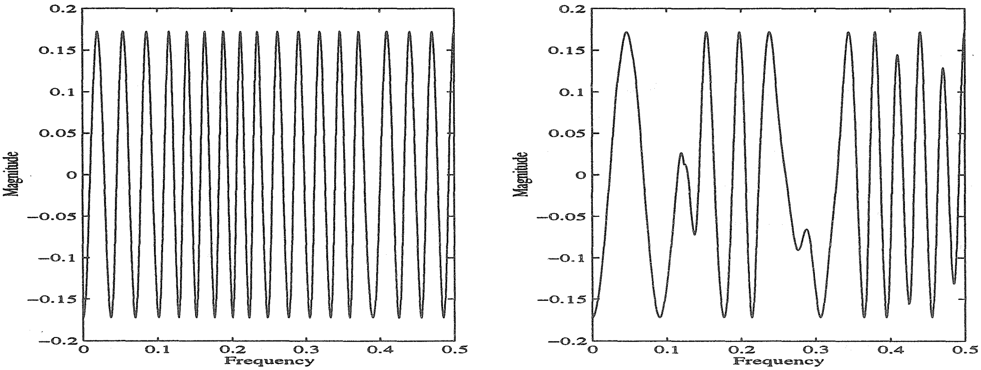

FIG. 5.3. *Error function for diagonal and off-diagonal elements*

[3] C. GONZAGA AND E. POLAK, On constraint dropping schemes and optimality functions for a class of outer approximations algorithms, *SIAM Journal of Optimization and Control*, 17 (1979), pp. 477–493.

[4] L. J. KARAM AND J. H. MCCLELLAN, Complex Chebyshev approximation for FIR filter design, *IEEE Transactions on Circuits and Systems—II: Analog and Digital Signal Processing*, 42 (1995), pp. 207-216.

[5] D. Q. MAYNE, E. POLAK, AND R. TRAHAN, An outer approximations algorithm for computer aided design problems, *Journal of Optimization and Applications*, 28 (1979), pp. 231–252.

[6] J. H. MCCLELLAN AND T. W. PARKS, A unified approach to the design of optimum FIR linear-phase digital filters, *IEEE Transactions on Circuit Theory*, CT-20 (1973), pp. 697–701.

[7] T. W. PARKS AND J. H. MCCLELLAN, Chebyshev approximation for nonrecursive digital filters with linear phase, *IEEE Transactions on Circuit Theory*, CT-19 (1972), pp. 189–194.

[8] J. G. PROAKIS AND D. G. MANOLAKIS, *Digital Signal Processing*, Prentice-Hall, third ed., 1996.

[9] D. W. TUFTS AND J. T. FRANCIS, Designing digital low-pass filters—comparison of some methods and criteria, *IEEE Transactions on Audio and Electroacoustics*, AU-18 (1970), pp. 487–494.

[10] L. VANDENBERGHE AND S. BOYD, *SP, Software for Semidefinite Programming, Beta Version, User's Guide*, K. U. Leuven, Stanford University, Nov. 1994.