Multidimensional FIR filter design via trigonometric sum-of-squares optimization

Tae Roh, Bogdan Dumitrescu, and Lieven Vandenberghe*†

June 15, 2007

Abstract

We discuss a method for multidimensional FIR filter design via sum-of-squares formulations of spectral mask constraints. The sum-of-squares optimization problem is expressed as a semidefinite program with low-rank structure, by sampling the constraints using discrete cosine and sine transforms. The resulting semidefinite program is then solved by a customized primal-dual interior-point method that exploits low-rank structure. This leads to a substantial reduction in the computational complexity, compared to general-purpose semidefinite programming methods that exploit sparsity.

1 Introduction

A variety of one-dimensional FIR filter design problems can be expressed as convex optimization problems over real trigonometric polynomials, subject to spectral mask constraints (upper or lower bounds on the polynomial over intervals of the frequency axis). These optimization problems can be formulated as semidefinite programs (SDPs) using classical sum-of-squares (SOS) characterizations of nonnegative polynomials, and solved efficiently via interior-point methods for semidefinite programming [2, 7, 13, 32]. The extension of these techniques to multidimensional filter design poses several difficulties. First, sum-of-squares characterizations of multivariate positive trigonometric polynomials may require factors of arbitrarily high degree. To obtain a tractable optimization problem, a bound on the degrees of the factor polynomials has to be imposed. This results in a sufficient condition for the

*T. Roh (roh@ee.ucla.edu) and L. Vandenberghe (vandenbe@ee.ucla.edu) are with the Department of Electrical Engineering, University of California, Los Angeles. Their research was supported by NSF under grant ECS-0524663.

†B. Dumitrescu is with Tampere International Center for Signal Processing, Tampere University of Technology, P.O.BOX 553, FIN-33101, Tampere, Finland, e-mail: bogdand@cs.tut.fi, on leave from the Department of Automatic Control and Computers, "Politehnica" University of Bucharest, Romania. His work was supported by Academy of Finland, Project No. 213462 (Finnish Centre of Excellence Program (2006-2011)).

original nonnegativity or spectral mask constraint [9–11], analogous to sum-of-squares formulations of real multivariate polynomials [16,21]. The second difficulty stems from the large size of the semidefinite programming problems obtained from multivariate sum-of-squares programs. This is due to the presence of large matrix variables with dimensions that grow exponentially with the number of variables in the multivariate polynomials. Hence there is a need for specialized SDP algorithms that exploit structure in multivariate sum-of-squares optimization problems.

Most recent research on exploiting structure in semidefinite programming has focused on exploiting sparsity of the coefficient matrices [12, 19, 31]. These techniques are very useful for SDPs derived from sum-of-squares programs, and are included in several general-purpose semidefinite programming packages. An alternative approach based on exploiting dense rank-one structure was studied in [3, 17, 24] and found to be very well-suited for sum-of-squares optimization. Our goal in this paper is to extend the techniques proposed in [24] to multivariate trigonometric polynomials, and to compare their effectiveness in practice with general-purpose solvers that exploit sparsity. Although we concentrate on applications to the two-dimensional FIR filter design problems discussed in detail in [9], the results are also relevant to general sum-of-squares optimization based on trigonometric basis functions [18].

An abridged version of this paper was published in [25].

Notation. $\mathbb{Z}^d$ and $\mathbb{N}^d$ denote the sets of d-vectors of integers and natural numbers, respectively. For a vector x, we define $\mathbf{diag}(x)$ as the diagonal matrix with x_i as its ith diagonal entry. For a square matrix X, $\mathbf{diag}(X)$ is a vector with the ith entry X_{ii}. The matrix inequality $A \succ (\succeq) B$ denotes that $A - B$ is positive definite (semidefinite). $\mathbf{tr}(A)$ denotes the trace of a symmetric matrix A.

2 Trigonometric sums of squares

Let R be a d-variate trigonometric polynomial of degree $\mathbf{n} \in \mathbb{Z}^d$, with real symmetric coefficients $x_{\mathbf{k}} = x_{-\mathbf{k}}$:

$$R(\omega) = \sum_{\mathbf{k}=-\mathbf{n}}^{\mathbf{n}} x_{\mathbf{k}} e^{-j\mathbf{k}^T\omega}. \tag{1}$$

The sum is over all integer vectors $\mathbf{k}$ that satisfy $-\mathbf{n} \leq \mathbf{k} \leq \mathbf{n}$, where the inequalities between the vectors are interpreted elementwise. A fundamental result [11, Theorem 3.5], [8, Theorem 5.1], [8, Corollary 5.2] is that if R is positive on $[-\pi, \pi]^d$, then it can be expressed as a sum of squares of trigonometric polynomials,

$$R(\omega) = \sum_{\ell=1}^{r} |H_\ell(\omega)|^2, \tag{2}$$

where the trigonometric polynomials $H_\ell(\omega)$ are linear combinations of the monomials $e^{-j\mathbf{k}^T\omega}$ with exponents in the first orthant ($\mathbf{k} \in \mathbb{N}^d$), *i.e.*, H_ℓ takes the form

$$H_\ell(\omega) = \sum_{\mathbf{k}=0}^{\mathbf{n}_\ell} h_{\ell,\mathbf{k}} e^{-j\mathbf{k}^T\omega}.$$

Note that in general the degrees $\mathbf{n}_\ell = \deg H_\ell$ may be greater than $\mathbf{n}$ (elementwise).

2.1 Sum-of-squares formulation

We denote by $\mathbb{P}_\mathbf{n}$ the set of nonnegative trigonometric polynomials of degree $\mathbf{n}$, and by $\mathbb{P}_\mathbf{n}^\mathbf{m}$ the set of sum-of-squares trigonometric polynomials (2) with factors of degree $\mathbf{n}_\ell \leq \mathbf{m}$. If $\mathbf{m} \geq \mathbf{n}$ then

$$\mathbb{P}_\mathbf{n}^\mathbf{n} \subseteq \mathbb{P}_\mathbf{n}^\mathbf{m} \subseteq \mathbb{P}_\mathbf{n}. \tag{3}$$

Now consider an optimization problem in which we constrain the trigonometric polynomial (1) to be positive on $[-\pi, \pi]^d$. Imposing $R \in \mathbb{P}_\mathbf{n}^\mathbf{m}$ instead of $R \in \mathbb{P}_\mathbf{n}$ is only a sufficient condition, but, as we explain later, it makes the problem solvable by standard convex optimization tools (typically semidefinite programming). Moreover it is observed in practice that the solution of the sum-of-squares problem is often the optimal one, even when we take $\mathbf{m} = \mathbf{n}$ [9]. This idea is related to the sum-of-squares relaxations for polynomial optimization discussed in [16,21,22]. We therefore refer to $\mathbf{m}$ as the *relaxation degree*. In order to simplify the notation, we will use $\mathbf{m} = \mathbf{n}$ in the remainder of the paper. However, all the results also apply to higher relaxation degrees.

2.2 Spectral mask constraints

We next discuss how spectral mask constraints are handled within the framework of [9, 11]. Let $\mathcal{D}$ be an intersection of regions $\mathcal{D}_i$ defined by trigonometric polynomials (of some low degree) D_i:

$$\mathcal{D} = \bigcap_{i=1}^{\ell} \mathcal{D}_i, \tag{4}$$

where

$$\mathcal{D}_i = \{\omega \in [-\pi, \pi]^d \mid D_i(\omega) \geq 0\}. \tag{5}$$

Then, a trigonometric polynomial R that is positive on $\mathcal{D}$ can be expressed as a *weighted* sum-of-squares

$$R(\omega) = S_0(\omega) + \sum_{i=1}^{\ell} D_i(\omega) S_i(\omega), \tag{6}$$

where S_i, $i = 0, \ldots, \ell$, are sums of squares of trigonometric polynomials [9], [11, Theorem 4.11]. In practice, if we impose $S_0 \in \mathbb{P}_\mathbf{n}^\mathbf{n}$, and $S_i \in \mathbb{P}_{\mathbf{n}_i}^{\mathbf{n}_i}$, where $\mathbf{n}_i = \mathbf{n} - \deg(D_i)$, then (6) provides a sufficient condition for nonnegativity on $\mathcal{D}$.

We can also consider spectral mask regions $\mathcal{D}$ described as a union

$$\mathcal{D} = \bigcup_{i=1}^{\ell} \mathcal{D}_i. \tag{7}$$

A trigonometric polynomial R positive on $\mathcal{D}$ must satisfy

$$R(\omega) = S_{i0}(\omega) + D_i(\omega) S_{i1}(\omega), \quad i = 1, \ldots, \ell, \tag{8}$$

where S_{ij} are sums of squares. This again is a sufficient condition if we limit the degrees of the sum-of-squares factors S_{ij}.

Although these formulations of spectral masks as intersections or unions of sets (5) appear to be limited, many practical two-dimensional spectral mask regions with simple geometrical shapes (rectangle, rhombus, fan) can be constructed [9, 11].

2.3 Two-dimensional FIR filter design as SOS program

Let H be the frequency response of a two-dimensional linear-phase FIR filter with filter order $\mathbf{n} = (n_1, n_2)$:

$$H(\omega) = \sum_{\mathbf{k}=-\mathbf{n}}^{\mathbf{n}} h_{\mathbf{k}} e^{-j\mathbf{k}^T\omega},$$

with real filter coefficients $h_{\mathbf{k}} = h_{-\mathbf{k}}$. We wish to determine the filter coefficients $h_{\mathbf{k}}$ that maximize the attenuation δ_{s} in the stopband $\mathcal{D}_{\mathrm{s}}$ for a given maximum allowable ripple (δ_{p}) in the passband $\mathcal{D}_{\mathrm{p}}$. The optimization problem is

$$\begin{array}{ll} \text{minimize} & \delta_{\mathrm{s}} \\ \text{subject to} & |1 - H(\omega)| \leq \delta_{\mathrm{p}}, \quad \omega \in \mathcal{D}_{\mathrm{p}}, \\ & |H(\omega)| \leq \delta_{\mathrm{s}}, \quad \omega \in \mathcal{D}_{\mathrm{s}}, \end{array} \tag{9}$$

where the scalar δ_{s} and the filter coefficients $h_{\mathbf{k}}$ are the problem variables. The constraints are equivalent to

$$R_1(\omega) = H(\omega) - 1 + \delta_{\mathrm{p}} \geq 0, \quad \omega \in \mathcal{D}_{\mathrm{p}} \tag{10}$$

$$R_2(\omega) = 1 - H(\omega) + \delta_{\mathrm{p}} \geq 0, \quad \omega \in \mathcal{D}_{\mathrm{p}} \tag{11}$$

$$R_3(\omega) = H(\omega) + \delta_{\mathrm{s}} \geq 0, \quad \omega \in \mathcal{D}_{\mathrm{s}} \tag{12}$$

$$R_4(\omega) = H(\omega) - \delta_{\mathrm{s}} \geq 0, \quad \omega \in \mathcal{D}_{\mathrm{s}}. \tag{13}$$

If the passband and stopband are defined as (4) or (7), then we can replace each positive polynomial R_i by a weighted sum-of-squares expression (5) or (8) (or a combination of the two). Limiting the degrees of the sum-of-squares polynomials to $\mathbf{n}$ then gives sufficient conditions for feasibility. We call the resulting optimization problem a *sum-of-squares program*. In section 3, we show how this sum-of-squares program can be solved via semidefinite programming.

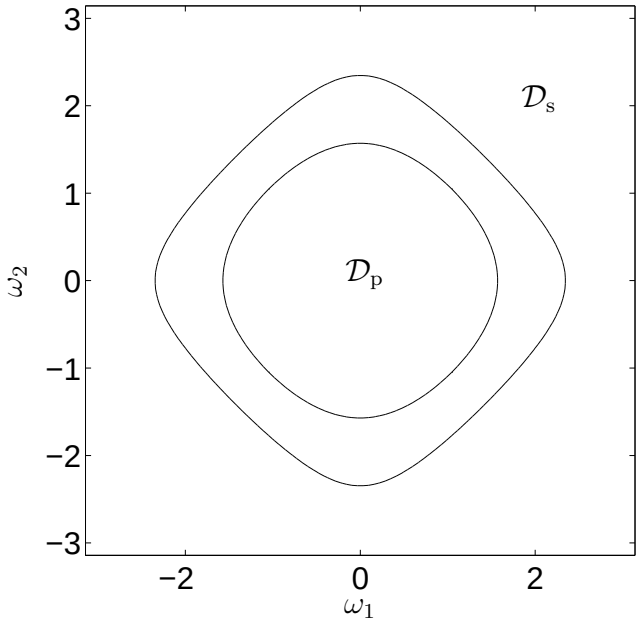

Figure 1: The spectral masks described by the sets (14) with $c_\mathrm{p} = 1$ and $c_\mathrm{s} = 0.3$.

Example We consider a simple 2-D lowpass filter problem with the following spectral mask constraints. Let

$$D_\mathrm{p}(\omega) = \cos(\omega_1) + \cos(\omega_2) - c_\mathrm{p}, \qquad D_\mathrm{s}(\omega) = c_\mathrm{s} - \cos(\omega_1) - \cos(\omega_2).$$

and define the spectral masks $\mathcal{D}_\mathrm{p}$ and $\mathcal{D}_\mathrm{s}$ as

$$\mathcal{D}_\mathrm{p} = \{\omega \in [-\pi, \pi]^2 \mid D_\mathrm{p}(\omega) \geq 0\}, \quad \mathcal{D}_\mathrm{s} = \{\omega \in [-\pi, \pi]^2 \mid D_\mathrm{s}(\omega) \geq 0\}. \tag{14}$$

Figure 1 illustrates these two regions. The sum-of-squares program for this filter is

$$\begin{array}{ll} \text{minimize} & \delta_\mathrm{s} \\ \text{subject to} & H(\omega) - 1 + \delta_\mathrm{p} = S_{10}(\omega) + D_\mathrm{p}(\omega) S_{11}(\omega) \\ & 1 - H(\omega) + \delta_\mathrm{p} = S_{20}(\omega) + D_\mathrm{p}(\omega) S_{21}(\omega) \\ & H(\omega) + \delta_\mathrm{s} = S_{30}(\omega) + D_\mathrm{s}(\omega) S_{31}(\omega) \\ & H(\omega) - \delta_\mathrm{s} = S_{40}(\omega) + D_\mathrm{s}(\omega) S_{41}(\omega) \\ & S_{i0} \in \mathbb{P}_{\mathbf{n}}, \quad S_{i1} \in \mathbb{P}_{\mathbf{n-1}}, \quad i = 1, \ldots, 4. \end{array} \tag{15}$$

(The symbol $\mathbf{1}$ in $\mathbb{P}_{\mathbf{n-1}}$ denotes the vector of ones.) The variables are the coefficients of trigonometric polynomials H, the sum-of-squares polynomials S_{ik}, and the stopband attenuation δ_s. Figure 2 shows the optimal solution of (15), with the design parameters $c_\mathrm{p} = 1$, $c_\mathrm{s} = 0, 3$, $\delta_\mathrm{p} = 0.05$, and 2-D filter order $\mathbf{n} = (n_1, n_2) = (11, 11)$. The optimal attenuation is 69 dB.

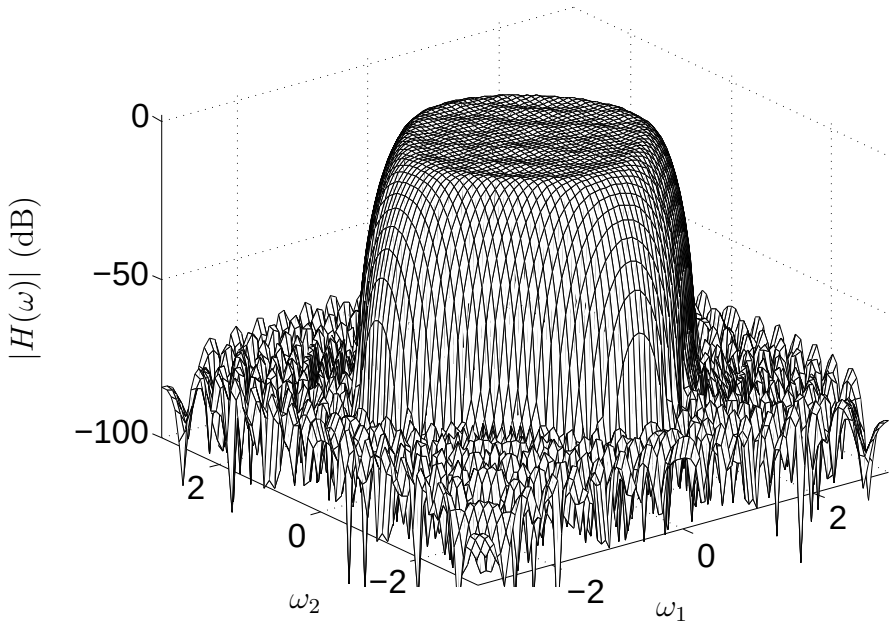

Figure 2: Solution of the lowpass filter problem (15) with $n_1 = n_2 = 11$ and passband and stopband shown in Figure 1.

3 Semidefinite programming formulations

A semidefinite program (SDP) is an optimization problem of the form

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(QX) + q^T y \\ \text{subject to} & \mathcal{A}(X) + By = c \\ & X \succeq 0. \end{array} \tag{16}$$

The unknowns are the matrix $X \in \mathbb{S}^m$ (the symmetric matrices of order m) and vector $y \in \mathbb{R}^p$. The problem parameters are $Q \in \mathbb{S}^m$, $B \in \mathbb{R}^{s\times p}$, $c \in \mathbb{R}^p$, and a linear mapping $\mathcal{A} : \mathbb{S}^m \to \mathbb{R}^s$. The inequality $X \succeq 0$ means that X is positive semidefinite. The problem (16) is called a *standard form* SDP (or more precisely, a standard form SDP with *free variables*). In practice it is very common that the variable X is block-diagonal, and its diagonal blocks X_k can be interpreted as separate smaller matrix variables.

In the applications considered in this paper, the mapping $\mathcal{A}$ has a special 'block-diagonal' structure, and the SDP can be written as

$$\begin{array}{ll} \text{minimize} & \sum_{k=1}^L \mathbf{tr}(Q_k X_k) + q^T y \\ \text{subject to} & \mathcal{A}_k(X_k) + B_k y = c_k, \quad k = 1, \ldots, L \\ & X_k \succeq 0, \quad k = 1, \ldots, L. \end{array} \tag{17}$$

Each of the matrix variables $X_k \in \mathbb{S}^{m_k}$ in this problem appears in only one of the L equality constraints, and $\mathcal{A}_k$ is a linear mapping from $\mathbb{S}^{m_k}$ to $\mathbb{R}^{s_k}$. The constraints are coupled via the variable y. In addition each variable X_k can itself be block-diagonal.

In this section we discuss two possible SDP formulations of the sum-of-squares programs discussed in Section 2. Although the two formulations are mathematically equivalent, they have different properties when it comes to numerical implementation of interior-point algorithms.

3.1 SDP formulation of sum-of-squares constraint

Consider the expression (2) with factors H_ℓ of degree $\mathbf{n}$, and assume for now that the elements of $\mathbf{n}$ are even. The trigonometric polynomial H_ℓ can be written as

$$\begin{aligned} H_\ell(\omega) &= e^{-j\mathbf{n}^T\omega/2} \sum_{0\leq\mathbf{k}\leq\mathbf{n}} h_{\ell,\mathbf{k}} \left(\cos((\mathbf{k}-\mathbf{n}/2)^T\omega) - j\sin((\mathbf{k}-\mathbf{n}/2))^T\omega)\right) \\ &= e^{-j\mathbf{n}^T\omega/2} \sum_{-\mathbf{n}/2\leq\mathbf{k}\leq\mathbf{n}/2} h_{\ell,\mathbf{k}+\mathbf{n}/2} \left(\cos(\mathbf{k}^T\omega) - j\sin(\mathbf{k}^T\omega)\right) \\ &= e^{-j\mathbf{n}^T\omega/2} \left(h_{\ell,\mathbf{n}/2} + \sum_{\mathbf{k}\in\mathcal{J}_d} (h_{\ell,\mathbf{k}+\mathbf{n}/2} + h_{\ell,-\mathbf{k}+\mathbf{n}/2})\cos(\mathbf{k}^T\omega) \right. \\ &\qquad \left. + j\sum_{\mathbf{k}\in\mathcal{J}_d} (h_{\ell,\mathbf{k}+\mathbf{n}/2} - h_{\ell,-\mathbf{k}+\mathbf{n}/2})\sin(\mathbf{k}^T\omega) \right), \end{aligned}$$

where $\mathcal{J}_d$ is a subset of $\mathbb{Z}^d$ that satisfies

$$\mathcal{J}_d \cup -\mathcal{J}_d = \{\mathbf{k}\in\mathbb{Z}^d \mid \mathbf{k}\neq 0, -\mathbf{n}/2 \leq \mathbf{k} \leq \mathbf{n}/2\}, \qquad \mathcal{J}_d \cap -\mathcal{J}_d = \emptyset.$$

For example, for $d = 2$, one can take

$$\mathcal{J}_d = \{(k_1,k_2)\in\mathbb{Z}^2 \mid -n_1/2 \leq k_1 \leq n_1/2, 0 < k_2 \leq n_2/2 \text{ or } 0 < k_1 \leq n_1/2, k_2 = 0\}.$$

We see that the trigonometric polynomial $\tilde{H}_\ell(\omega) = e^{j\mathbf{n}^T\omega/2}H_\ell(\omega)$ can be written as

$$\tilde{H}_\ell(\omega) = a_\ell^T v(\omega) + jb_\ell^T w(\omega), \tag{18}$$

where $v(\omega)$ is a vector of the trigonometric basis functions $\cos(\mathbf{k}^T\omega)$ for $\mathbf{k}\in\mathcal{J}_d\cup\{0\}$, $w(\omega)$ is a basis vector of the functions $\sin(\mathbf{k}^T\omega)$ for $\mathbf{k}\in\mathcal{J}_d$, and a_ℓ, b_ℓ are real vectors that depend linearly on the coefficients of H_ℓ. The dimensions of the vectors v and w are

$$M_v = (M+1)/2, \qquad M_w = (M-1)/2,$$

respectively, where $M = \prod_{i=1}^d (n_i+1)$.

Since $|\tilde{H}_\ell(\omega)| = |H_\ell(\omega)|$, it follows from (2) and (18) that

$$\begin{aligned} R(\omega) &= \sum_{\ell=1}^r |a_\ell^T v(\omega) + jb_\ell^T w(\omega)|^2 \\ &= v(\omega)^T X_1 v(\omega) + w(\omega)^T X_2 w(\omega), \end{aligned}$$

where $X_1 = \sum_\ell a_\ell a_\ell^T$ and $X_2 = \sum_\ell b_\ell b_\ell^T$. We can therefore conclude that $R \in \mathbb{P}^{\mathbf{n}}_{\mathbf{n}}$ if and only if there exist positive semidefinite matrices $X_1 \in \mathbb{S}^{M_v}$, $X_2 \in \mathbb{S}^{M_w}$ such that

$$R(\omega) = v(\omega)^T X_1 v(\omega) + w(\omega)^T X_2 w(\omega), \quad X_1 \succeq 0, \quad X_2 \succeq 0. \tag{19}$$

Similar expressions hold if $\mathbf{n}$ contains at least one odd integer, with slightly different $v(\omega)$ and $w(\omega)$, and M_v and M_w equal to $M/2$. More details and a discussion of other parity cases can be found in [10].

The expression (19) is readily extended to weighted sums of squares. The constraint (6), for example, can be written as

$$\begin{aligned} R(\omega) &= v_0(\omega)^T X_{01} v_0(\omega) + w_0(\omega)^T X_{02} w_0(\omega) \\ &\quad + \sum_{i=1}^{\ell} D_i(\omega) \left(v_i(\omega)^T X_{i1} v_i(\omega) + w_i(\omega)^T X_{i2} w_i(\omega) \right) \end{aligned} \tag{20}$$

with $X_{i1} \succeq 0$, $X_{i2} \succeq 0$, where v_i and w_i are vectors of the trigonometric basis functions $\cos(\mathbf{k}^T\omega)$ and $\sin(\mathbf{k}^T\omega)$, respectively.

The constraints (19) and (20) are in standard SDP form; they are linear equalities between the coefficients of the trigonometric polynomial R and positive semidefinite matrices X_1, X_2 (resp., X_{i1}, X_{i2} for $i = 0, \ldots, l$). The equality constraints are expressed as an infinite number of linear equations, and in order to handle them by interior-point methods, we need to formulate them as finite sets of equations. In the next sections we outline two methods for this.

3.2 Gram-pair parametrizations

Expanding the righthand side of the equality in (19) gives

$$R(\omega) = \sum_{\mathbf{i},\mathbf{l} \in \mathcal{J}_d \cup \{0\}} X_{1,\mathbf{il}} \cos(\mathbf{i}^T\omega) \cos(\mathbf{l}^T\omega) + j \sum_{\mathbf{i},\mathbf{l} \in \mathcal{J}_d} X_{2,\mathbf{il}} \sin(\mathbf{i}^T\omega) \sin(\mathbf{l}^T\omega)$$

where $X_{1,\mathbf{il}}$ is the entry of X_1 that is multiplied with $\cos(\mathbf{i}^T\omega)\cos(\mathbf{l}^T\omega)$ in the quadratic form $v(\omega)^T X_1 v(\omega)$, and $X_{2,\mathbf{il}}$ is the entry of X_2 that is multiplied with $\sin(\mathbf{i}^T\omega)\sin(\mathbf{l}^T\omega)$ in the quadratic form $w(\omega)^T X_2 w(\omega)$. Using elementary trigonometric trigonometric identities this expression can be written as

$$\begin{aligned} R(\omega) = \frac{1}{2} &\sum_{\mathbf{i},\mathbf{l} \in \mathcal{J}_d \cup \{0\}} X_{1,\mathbf{il}} \left(\cos((\mathbf{i}+\mathbf{l})^T\omega) + \cos((\mathbf{i}-\mathbf{l})^T\omega) \right) \\ + \frac{1}{2} &\sum_{\mathbf{i},\mathbf{l} \in \mathcal{J}_d} X_{2,\mathbf{il}} \left(-\cos((\mathbf{i}+\mathbf{l})^T\omega) + \cos((\mathbf{i}-\mathbf{l})^T\omega) \right). \end{aligned} \tag{21}$$

We can now convert the equality in (19) to a finite set of equations by in the same basis as the righthand side of (21), *i.e.*,

$$R(\omega) = x_0 + 2 \sum_{\mathbf{k} \in \mathcal{J}_d} x_{\mathbf{k}} \cos(\mathbf{k}^T\omega), \tag{22}$$

and equating the coefficients of the same basis functions on both sides of (21). This gives a set of linear equations in $x_{\mathbf{k}}$, X_1, and X_2. Together with the inequality constraints in (19), this allows us to write (19) in the form

$$x_{\mathbf{k}} = \mathbf{tr}(T_{\mathbf{k},1}X_1) + \mathbf{tr}(T_{\mathbf{k},2}X_2), \quad \mathbf{k} \in \mathcal{J}_d \cup \{0\}, \qquad X_1 \succeq 0, \qquad X_2 \succeq 0. \tag{23}$$

The coefficient matrices $T_{\mathbf{k},1}$ and $H_{\mathbf{k},2}$ are very sparse and can be built efficiently [10, 11]. We refer to (23) as the *Gram-pair* parametrization of the sum-of-squares polynomial R.

The relation (23) differs from the Gram parametrization used in [9, 23], $x_{\mathbf{k}} = \mathbf{tr}(\Theta_{\mathbf{k}}X)$, where $X \succeq 0$ and $\Theta_{\mathbf{k}}$ is a Kronecker product of elementary Toeplitz matrices. The size of the parameter matrix X is $M \times M$, *i.e.*, about twice as large as the matrices X_1 and X_2 in (23). This makes the Gram-pair parametrization (23) more efficient, as confirmed by experimental tests for 2-D polynomials [10].

3.3 Discrete transform parametrization

As an alternative to the Gram-pair parametrization, we can pass from (19) to a finite set of equations by specifying that the two sides of the equation must agree at a sufficient number (say, N) of distinct values ω_i. This formulation was recently proposed in [17, 24] as a basis of fast algorithms for sum-of-squares programs.

Applied to (19) this gives

$$R(\omega_i) = v(\omega_i)^T X_1 v(\omega_i) + w(\omega_i)^T X_2 w(\omega_i), \quad i = 1, \ldots, N, \qquad X_1 \succeq 0, \qquad X_2 \succeq 0,$$

or in matrix form,

$$Fx = \mathbf{diag}(VX_1V^T + WX_2W^T), \qquad X_1 \succeq 0, \qquad X_2 \succeq 0. \tag{24}$$

Here, x is a vector that contains the coefficients of R in (22), and $Fx = (R(\omega_1), \ldots, R(\omega_N))$ are the values of R at the sample points ω_i. The matrices V and W have rows $v(\omega_i)^T$ and $w(\omega_i)^T$, respectively. The matrices F, V, and W represent discrete transforms that map the coefficients of trigonometric polynomials to their values at the points ω_i. For the sample points we use a rectangular grid in $[-\pi, \pi]^d$

$$\omega_i = \frac{2\pi k}{2n_i + 1}, \quad k = -n_i, \ldots, n_i, \quad i = 1, \ldots, d \tag{25}$$

(see [6, pp. 69]). The choice of grid is important, since orthogonality of discrete transforms depends on the sample locations. For $d = 2$ this yields $(2n_1 + 1)(2n_2 + 1)$ sample points. However, due to our assumption of real symmetry (which translates to the symmetry about the origin of the sample grid), we require only (roughly) half of the samples, *i.e.*, $N = (2n_1 + 1)(n_2 + 1) - n_1$ for $d = 2$. Figure 3 shows an example of a possible sampling grid for $d = 2$ and $n_1 = n_2 = 5$.

For a uniform grid of sample frequencies, the matrices V and W are discrete cosine transform and discrete sine transform matrices. If we take into account the symmetry of the grid and of the polynomial coefficients $x_{\mathbf{k}}$, then F can be constructed from a DCT matrix.

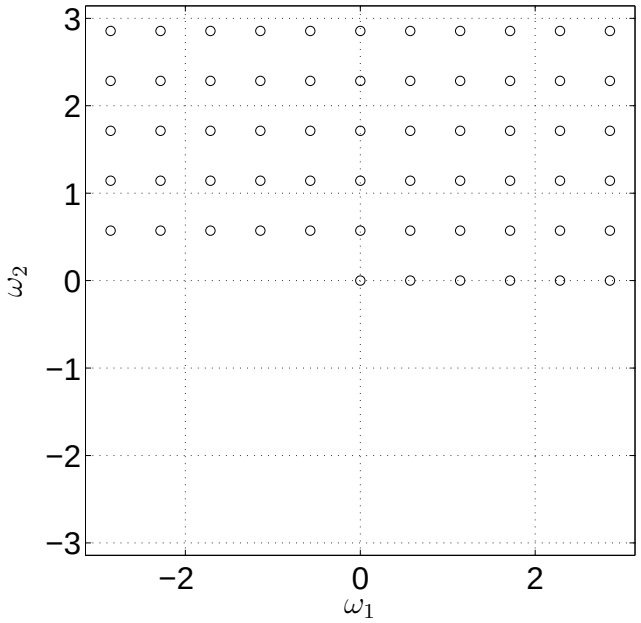

Figure 3: An example of the (square) sampling grid (25) for $d = 2$ and $n_1 = n_2 = 5$. There are 61 sampling points.

Note that the constraint (24) is equivalent to (23), and in particular, the matrices X_1 and X_2 in the two expressions are the same. In (23) the linear relation between x and the matrices X_1 and X_2 is expressed as inner products with sparse matrices; in (26), it is expressed as linear combinations of inner products with rank-one matrices $v(\omega_i)v(\omega_i)^T$ and $w(\omega_i)w(\omega_i)^T$. Although the formulation (26) involves dense matrices, the presence of the **diag** operator leads to an algorithm with a significant reduction in computational complexity (see section 4).

From the sample value $y = Fx$ the coefficient vector x can be obtained via the corresponding 'inverse' transform $x = Gy$, which maps samples of R to its coefficients and satisfies $GF = I$. Thus, an alternative to (24) is

$$x = G\,\mathbf{diag}(VX_1V^T + WX_2W^T), \qquad X_1 \succeq 0, \qquad X_2 \succeq 0. \tag{26}$$

When the number of sample points is equal to the number of unknown coefficients in R, the matrices F and G are square and there is no particular advantage in choosing (26) over (24). However, there exist applications in which a certain set of filter coefficients are deliberately omitted (*e.g.*, see [30, Section 12.8]), and F may be rectangular with more rows than columns. In such cases the formulation (24) is less attractive than (26), since in (24) the number of equality constraints increases linearly with the number of sample points. In (24), on the other hand, the number of equalities is constant, and only the column dimension of G and the row dimensions of V and W increase with sampling size.

Similar observations apply to the SDP formulation of the weighted sum of squares (20). By sampling the two sides of the equality and mapping the sample values back to the

coefficients of R we obtain an expression of the form

$$x = G\left(\mathbf{diag}(V_0 X_{01} V_0^T + W_0 X_{02} W_0^T) + \sum_{i=1}^{\ell} d_i \circ \mathbf{diag}(V_i X_{i1} V_i^T + W_i X_{i2} W_i^T)\right), \tag{27}$$

where $X_{i1} \succeq 0$, $X_{i2} \succeq 0$, $i = 0, \ldots, l$. Here '$\circ$' denotes the Hadamard (component-wise) product. V_i and W_i are discrete cosine and sine transform matrices and d_i are vectors of samples of the trigonometric polynomials D_i.

3.4 Two-dimensional filter design

As an example, we express the 2-D lowpass filter design problem (15) as an SDP. First, each weighted sum-of-squares constraint is parametrized with Gram matrices as shown in (20). For example, the first constraint of (15) is expressed as

$$\begin{aligned} & H(\omega) - 1 + \delta_{\mathrm{p}} \\ &= v_0(\omega)^T X_{01} v_0(\omega) + w_0(\omega)^T X_{02} w_0(\omega) + D_{\mathrm{p}}(\omega)\left(v_1(\omega)^T X_{11} v_1(\omega) + w_1(\omega)^T X_{12} w_1(\omega)\right). \end{aligned}$$

By applying forward and inverse transforms as in (27) we obtain

$$h - (1-\delta_{\mathrm{p}})e_0 = G\left(\mathbf{diag}(V_0 X_{01} V_0^T + W_0 X_{02} W_0^T) + d_p \circ \mathbf{diag}(V_1 X_{i1} V_1^T + W_1 X_{i2} W_1^T)\right),$$

where h contains the coefficients of H, d_{p} is the vector of samples of the polynomial D_{p}, and e_0 is the first unit vector. Repeating this process for the remaining constraints of (15), we obtain an equivalent SDP

$$\begin{array}{ll} \text{minimize} & \delta_{\mathrm{s}} \\ \text{subject to} & h - (1-\delta_{\mathrm{p}})e_0 = G\mathcal{H}_0(X_1) + G_{\mathrm{p}}\mathcal{H}_1(X_2) \\ & -h + (1+\delta_{\mathrm{p}})e_0 = G\mathcal{H}_0(X_3) + G_{\mathrm{p}}\mathcal{H}_1(X_4) \\ & h + \delta_{\mathrm{s}} e_0 = G\mathcal{H}_0(X_5) + G_{\mathrm{s}}\mathcal{H}_1(X_6) \\ & -h + \delta_{\mathrm{s}} e_0 = G\mathcal{H}_0(X_7) + G_{\mathrm{s}}\mathcal{H}_1(X_8) \\ & X_i \succeq 0, \quad i = 1, \ldots, 8. \end{array} \tag{28}$$

where, for a block-diagonal matrix Y with two diagonal blocks Y_1 and Y_2 of appropriate dimensions,

$$\mathcal{H}_k(Y) = \mathbf{diag}(V_k Y_1 V_k^T + W_k Y_2 W_k^T), \quad k = 0, 1,$$

$G_{\mathrm{p}} = G\,\mathbf{diag}(d_{\mathrm{p}})$ and $G_{\mathrm{s}} = G\,\mathbf{diag}(d_{\mathrm{s}})$. The variables in the SDP (28) are the $(2n_1+1)(n_2+1) - n_1$ coefficients in h, and eight block-diagonal matrices X_k, each with two blocks of size roughly $n_1 n_2/2$. Each equality constraint has dimension $(2n_1+1)(n_2+1) - n_1$.

3.5 Bounded Real Lemma

As another example we consider now a nonlinear phase filter

$$G(\omega) = \sum_{\mathbf{k}=0}^{\mathbf{n}} g_{\mathbf{k}} e^{-j\mathbf{k}^T\omega}.$$

Its frequency response $G(\omega)$ is complex and thus a direct formulation of design problems in terms of positive polynomials, as in the previous subsection, is not possible. Instead, we can characterize as an inequality

$$|G(\omega)| < \gamma, \ \forall \omega \in \mathcal{D}, \tag{29}$$

where $\mathcal{D}$ is defined as in (4)-(5). In [9], this Bounded Real Lemma (BRL) inequality was relaxed to a linear matrix inequality (LMI) using the Gram matrix parametrization. Here, we present a new LMI.

As in (18), we write

$$\tilde{G}_(\omega) = e^{j\mathbf{n}^T\omega/2} G(\omega) = (\Gamma_v g)^T v(\omega) + j(\Gamma_w g)^T w(\omega), \tag{30}$$

where g is a vector containing the coefficients of G and Γ_v, Γ_w are constant matrices (easy to derive). If (29) holds, then the polynomial $R(\omega) = \gamma^2 - |H(\omega)|^2$ is positive and thus admits a representation (6). Following the same reasoning as in [9], using this representation, a majorization result and a Schur complement argument, we can prove the following.

Bounded Real Lemma. The inequality (29) holds for some $\gamma > 0$ and $\mathcal{D}$ defined as in (4)-(5), if there exist matrices $X_{i0} \succeq 0$, $X_{i1} \succeq 0$, $i = 0, \ldots, l$, such that

$$\gamma^2 e_0 = G\bigg(\mathbf{diag}(V_0 X_{01} V_0^T + W_0 X_{02} W_0^T) + \sum_{i=1}^{\ell} d_i \circ \mathbf{diag}(V_i X_{i1} V_i^T + W_i X_{i2} W_i^T) \bigg) \tag{31}$$

i.e., the equality (27) holds for the positive 'polynomial' γ^2, and

$$\begin{bmatrix} X_{01} & \Gamma_v g \\ (\Gamma_v g)^T & 1 \end{bmatrix} \succeq 0, \qquad \begin{bmatrix} X_{02} & \Gamma_w g \\ (\Gamma_w g)^T & 1 \end{bmatrix} \succeq 0. \tag{32}$$

This BRL can be used to design nonlinear phase FIR filters, as described in [9]. The advantages over the BRL from [9] are the smaller size of the variable matrices (inherited from the Gram-pair parametrization) and the applicability of fast algorithms, as shown in the next section.

4 Primal-dual interior-point methods

Primal-dual interior-point methods for semidefinite programming simultaneously solve the SDP (16) and the corresponding dual problem

$$\begin{array}{ll} \text{maximize} & c^T z \\ \text{subject to} & \mathcal{A}^{\text{adj}}(z) \preceq Q \\ & B^T z = q. \end{array}$$

The variable in the dual problem is the vector $z \in \mathbb{R}^s$. The mapping $\mathcal{A}^{\rm adj}$ is the adjoint of $\mathcal{A}$, *i.e.*, it is the mapping from $\mathbb{R}^s$ to $\mathbb{S}^m$ that satisfies

$$u^T \mathcal{A}(V) = \mathbf{tr}(V \mathcal{A}^{\rm adj}(u))$$

for all $u \in \mathbb{R}^s$ and $V \in \mathbb{S}^m$. The dual of the SDP (17) is

$$\begin{array}{ll} \text{maximize} & \sum_{k=1}^L c_k^T z_k \\ \text{subject to} & \mathcal{A}_k^{\rm adj}(z_k) \preceq Q_k, \quad k = 1, \ldots, L \\ & \sum_{k=1}^L B_k^T z_k = q, \end{array}$$

and has L variables $z_k \in \mathbb{R}^{s_k}$.

4.1 Newton equations

To form an estimate of the cost of solving an SDP via an interior-point method, it is sufficient to know that the number of iterations is typically between 10 and 50, for a very wide range of problem sizes, and that the main step in each iteration is the solution of a large set of linear equations to compute primal and dual search directions. These equations are often referred to as the *Newton equation* because they can be viewed as a linearization of the nonlinear equations that characterize the central path. The construction and solution of the Newton equations are the most time-consuming part in an iteration, and we therefore limit our discussion of interior-point methods to the Newton equations. (For more details on different types of primal-dual SDP methods, see, for example, [1, 14, 15, 27].)

For a popular class of algorithms, including those implemented in the software packages SeDuMi [26] and SDPT3 [27–29], the Newton equations take the form

$$\begin{aligned} -T^{-1} \Delta X T^{-1} + \mathcal{A}^{\rm adj}(\Delta z) &= R_1 \\ \mathcal{A}(\Delta X) + B \Delta y &= r_2 \\ B^T \Delta z &= r_3. \end{aligned}$$

The matrix T is a positive definite matrix, with a different value in each iteration. Eliminating ΔX from the first equation gives

$$\left[\begin{array}{cc} H & B \\ B^T & 0 \end{array} \right] \left[\begin{array}{c} \Delta z \\ \Delta y \end{array} \right] = \left[\begin{array}{c} r_4 \\ r_3 \end{array} \right], \tag{33}$$

where $r_4 = r_2 + \mathcal{A}(T R_1 T)$ and H is defined by

$$H \Delta z = \mathcal{A}(T \mathcal{A}^{\rm adj}(\Delta z) T). \tag{34}$$

The solution of the Newton equations therefore involves three steps: the construction of the matrix H from T, the solution of the linear equations (33) to find Δz and Δy, and finally the computation of $\Delta X = T(\mathcal{A}^{\rm adj}(\Delta z) - R_1)T$. In many applications, the first step is the

most expensive of the three. It is also the part of the algorithm that provides the best opportunities for exploiting structure in the mapping $\mathcal{A}$.

In a similar way, we can solve the Newton equations for the SDP (17)

$$\begin{aligned} -T_k^{-1}\Delta X_k T_k^{-1} + \mathcal{A}_k^{\rm adj}(\Delta z_k) &= R_k, \quad k = 1, \ldots, L \\ \mathcal{A}_k^{\rm adj}(\Delta X_k) + B_k \Delta y &= r_{2k}, \quad k = 1, \ldots, L \\ \sum_{k=1}^{L} B_k^T \Delta z_k &= r_3, \end{aligned}$$

by eliminating the variables ΔX_k and solving

$$\left[\begin{array}{ccccc} H_1 & 0 & \cdots & 0 & B_1 \\ 0 & H_2 & \cdots & 0 & B_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & H_L & B_L \\ B_1^T & B_2^T & \cdots & B_L^T & 0 \end{array}\right] \left[\begin{array}{c} \Delta z_1 \\ \Delta z_2 \\ \vdots \\ \Delta z_L \\ \Delta y \end{array}\right] = \left[\begin{array}{c} r_{4,1} \\ r_{4,2} \\ \vdots \\ r_{4,L} \\ r_3 \end{array}\right]$$

where H_k is defined by

$$H_k \Delta z_k = \mathcal{A}_k(T_k \mathcal{A}_k^{\rm adj}(\Delta z_k) T_k).$$

4.2 Gram pair parametrization

General-purpose solvers assume that the mapping $\mathcal{A}$ in (16) is expressed as a vector of inner products with sparse symmetric coefficient matrices F_i:

$$\mathcal{A}(X) = \left[\begin{array}{c} \mathbf{tr}(F_1 X) \\ \vdots \\ \mathbf{tr}(F_s X) \end{array}\right], \qquad \mathcal{A}^{\rm adj}(z) = \sum_{i=1}^{s} z_i F_i. \tag{35}$$

The matrix H is then defined by

$$H_{ik} = \mathbf{tr}(T F_i T F_k), \quad i, k = 1, \ldots, s. \tag{36}$$

Let us assume for simplicity that the three problem dimensions m, p and s of the problem are of the same order (as is the case in the applications considered in this paper), *i.e.*, $m = O(p)$, $s = O(p)$. If the matrices F_i are dense, the cost of forming H is $O(p^4)$ and the cost of solving the equations (33) is $O(p^3)$. Although sparsity in the matrices F_i helps to lower the cost of constructing H, the cost of forming H usually still dominates the cost of an iteration.

For block diagonal SDPs (17), if

$$\mathcal{A}_k(X_k) = \left[\begin{array}{c} \mathbf{tr}(F_{k1} X_k) \\ \vdots \\ \mathbf{tr}(F_{k,m_k} X_k) \end{array}\right], \quad k = 1, \ldots, L, \tag{37}$$

with dense coefficient matrices and dimensions $m_k = O(p)$, $s_k = O(p)$, we obtain a cost estimate of $O(Lp^4)$ operations to construct the L matrices H_k, and $O(Lp^3)$ for solving the equations. Sparsity in the coefficient matrices can again reduce the cost of constructing H_k.

The Gram pair SDP formulation of sum-of-squares programs, based on (23), naturally leads to SDPs with linear mappings in the canonical form (37), with very sparse coefficient matrices F_{ki}.

4.3 Discrete transform parametrization

Next, we consider a class of structured SDPs (16) in which the mapping $\mathcal{A}$ is defined as

$$\mathcal{A}(X) = A\,\mathbf{diag}(CXC^T), \tag{38}$$

with $A \in \mathbb{R}^{s\times r}$, $C \in \mathbb{R}^{r\times m}$ and $r \ll ms$. The mapping $\mathcal{A}(x)$ is a linear combination of inner products of X with r rank-one matrices $c_i c_i^T$, where c_i^T is the ith row of C. This structure arises in many applications, in particular, SDP formulations of sum-of-squares programs. For example, it includes the discrete transform parameterizations of section 3.3.

It can be shown that the matrix H in (34) is equal to

$$H = A\left((CTC^T) \circ (CTC^T)\right) A^T. \tag{39}$$

(See [24].) If the dimensions s, m, r are all of the same order as p, then the cost of computing H is $O(p^3)$, much less than the $O(p^4)$ complexity of (36) for dense matrices.

For block-diagonal SDPs (17), with

$$\mathcal{A}_k(X_k) = A_k\,\mathbf{diag}(C_k X_k C_k^T), \quad k = 1, \ldots, L,$$

and $A_k \in \mathbb{R}^{s_k\times r_k}$, $C_k \in \mathbb{R}^{r_k\times m_k}$ we obtain a complexity of $O(Lp^3)$ per iteration, if we assume that s_k, r_k, m_k are $O(p)$.

Note that if $A = I$, the mapping $\mathcal{A}(X)$ is a vector of inner products with r rank-one matrices $c_i c_i^T$. This special case can be handled by the general-purpose solvers DSDP [3–5] and SDTP3 (ver. 4.0 beta) [27, 29]. Including a non-square matrix A is often useful, and allows us to handle the constraint (26), for example, with nonsquare G.

Example In the 2-D FIR lowpass filter design problem (28) with filter order $n_1 = n_2 = n$, we have $L = 4$, $p = O(n^2)$, $m_k = O(n^2)$, $s_k = O(n^2)$, $r_k = O(n^2)$. This gives a complexity of $O(n^6)$ per iteration. We can compare this with the complexity of the Gram pair SDP formulation, which is $O(n^8)$ if sparsity is not exploited. If we exploit sparsity it is substantially less (between $O(n^6)$ and $O(n^8)$), as we will see in the experiments of the next section.

5 Numerical results

5.1 Test problems

We will use three test examples. The first example is the simple 2-D lowpass filter design problem (15). The second example is a filter design problem, similar to (15) but with a more complex description of the passband and stopband regions. We consider a 2-D filter with a diamond-shaped passband region,

$$\mathcal{D}_{\rm p} = \bigcap_{i=1}^{3} \mathcal{D}_i, \qquad \mathcal{D}_{\rm s} = \bigcup_{i=4}^{6} \mathcal{D}_i, \tag{40}$$

where

$$\begin{aligned}
\mathcal{D}_1 &= \{\omega \in [-\pi,\pi]^2 \mid \cos(\omega_1+\omega_2) - c_{\rm p} \geq 0\} && (41)\\
\mathcal{D}_2 &= \{\omega \in [-\pi,\pi]^2 \mid \cos(\omega_1-\omega_2) - c_{\rm p} \geq 0\} && (42)\\
\mathcal{D}_3 &= \{\omega \in [-\pi,\pi]^2 \mid \cos\omega_1 + \cos\omega_2 \geq 0\} && (43)\\
\mathcal{D}_4 &= \{\omega \in [-\pi,\pi]^2 \mid c_{\rm s} - \cos(\omega_1+\omega_2) \geq 0\} && (44)\\
\mathcal{D}_5 &= \{\omega \in [-\pi,\pi]^2 \mid c_{\rm s} - \cos(\omega_1-\omega_2) \geq 0\} && (45)\\
\mathcal{D}_6 &= \{\omega \in [-\pi,\pi]^2 \mid -\cos\omega_1 - \cos\omega_2 \geq 0\} && (46)
\end{aligned}$$

and $c_{\rm p}$ and $c_{\rm s}$ are parameters defining the positions of the rhombi that are the passband and stopband edges; $\mathcal{D}_{\rm p}$ is an intersection of positive regions of three polynomials ((41) through (43)), and its SDP representation has 8 matrix variables. $\mathcal{D}_{\rm s}$, a union of three sets ((44) through (46)), leads to a set of SDP representations with 12 matrix variables. We take $n_1 = n_2 = n$. There are two positive polynomials associated with each of $\mathcal{D}_{\rm p}$ and $\mathcal{D}_{\rm s}$, giving this problem a total of 40 matrix variables of size roughly $n^2/2$. We use the design parameter values $c_{\rm p} = 0$, $c_{\rm s} = -0.7$ (see Figure 4) and $\delta_{\rm p} = 0.1$ for this problem instance. We can formulate an SDP that is similar to the problem (28), but with $L = 8$ constraints and 40 matrix variables. For the filter orders $n = 10$, the solution to the filter design with the constraints (40) produces the filter seen in Figure 5. The optimal stopband attenuation is 50 dB.

The third example demonstrates another spectral mask geometry that can be obtained from a set of low-degree trigonometric polynomials. For this instance, we consider a fan filter with the passband and stopband regions shown in Figure 6, and they are defined by

$$\mathcal{D}_{\rm p} = \mathcal{D}_1 \cap \mathcal{D}_2, \qquad \mathcal{D}_{\rm s} = \mathcal{D}_3 \cup \mathcal{D}_4, \tag{47}$$

where

$$\begin{aligned}
\mathcal{D}_1 &= \{\omega \in [-\pi,\pi]^2 \,|\, 2\cos\omega_1 - \cos\omega_2 - 1 \geq 0\},\\
\mathcal{D}_2 &= \{\omega \in [-\pi,\pi]^2 \mid \cos\omega_2 \geq 0\},
\end{aligned}$$

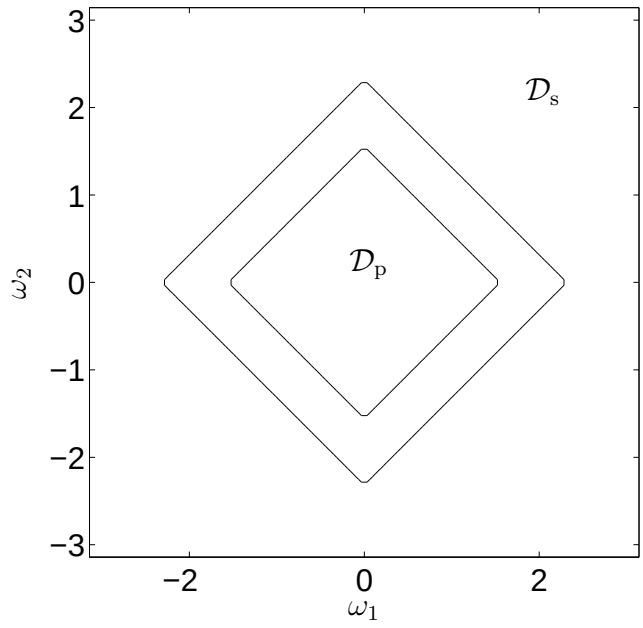

Figure 4: The spectral mask regions for the diamond filter in Figure 5.

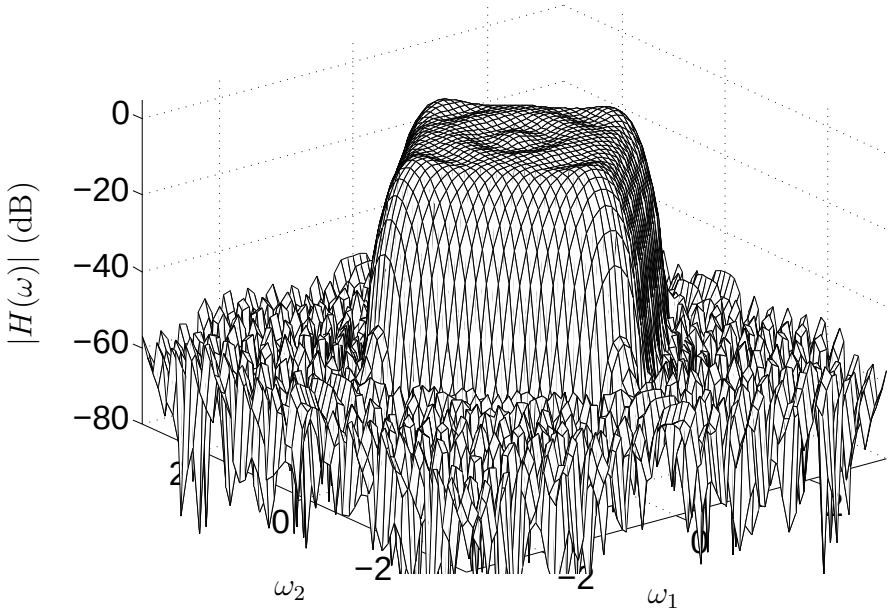

Figure 5: Solution of the diamond filter problem (40), with $n_1 = n_2 = 10$.

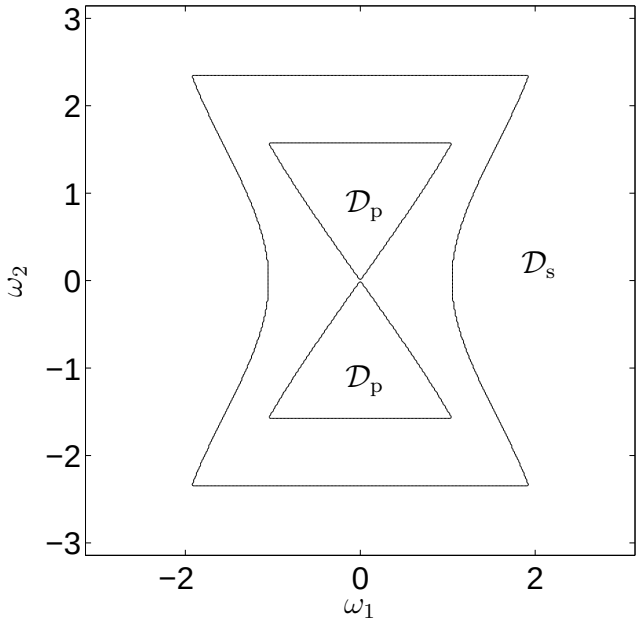

Figure 6: The spectral mask regions for the fan filter in Figure 7.

and

$$\begin{aligned}\mathcal{D}_3 &= \{\omega \in [-\pi,\pi]^2 \mid -2\cos\omega_1 + \cos\omega_2 \geq 0\},\\ \mathcal{D}_4 &= \{\omega \in [-\pi,\pi]^2 \mid -0.7 - \cos\omega_2 \geq 0\}.\end{aligned}$$

In this SDP problem, there are 28 semidefinite variables of size roughly $n_1 n_2/2$. With the design parameters $\delta_\mathrm{p} = 0.1$ and $n_1 = n_2 = 7$, we obtain the optimal attenuation of 33.6 dB. The corresponding filter response is shown in Figure 7.

5.2 Results

We solved the first two test problems for various filter orders using the two formulations of section 3. The sparse Gram-pair formulation (with constraints in the form (23)) is solved with the general-purpose SDP solvers (SeDuMi v. 1.1, DSDP v. 5.8, and SDPT3 v. 4.0 beta). (More precisely, the SDP from the Gram-pair formulation is put in the SeDuMi format. The SDP description is then converted to the DSDP and SDPT3 formats using the routines that are provided with these packages.) The discrete-transform formulation (with constraints in the form (26)) is solved using a Matlab implementation of a primal-dual path-following method similar to the algorithms used in SeDuMi and SDPT3, but with the Newton equations solved using the fast technique outlined in section 4.3. This Matlab method does not include many of the advanced features implemented in the general-purpose solvers (such as infeasibility detection, starting point selection, conjugate gradient refinement, etc.)[1]. It

[1]DSDP v. 5.8 and SDPT3 v. 4.0 beta support SDPs with low-rank coefficients, with certain limitations. First, DSDP does not admit free vector variable (the By term in (16)). In our experiments, we had to

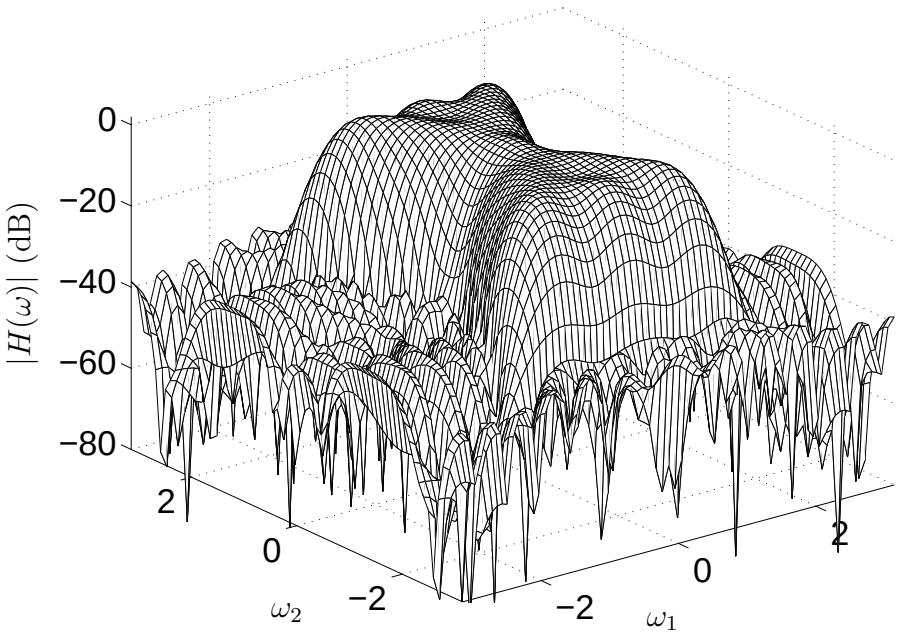

Figure 7: Solution of the diamond filter problem (47), with $n_1 = n_2 = 7$.

also does not take advantage of fast discrete transform algorithms when computing the matrix H in Newton equation. The experiments were run on an Intel Pentium-4 3GHz computer with 3GB RAM memory and Linux-version of MATLAB 7.1 (R14).

Table 1 shows the CPU time per iteration (in seconds) for each of the solvers, applied to the lowpass filter shown in Figure 2. Since the purpose of this experiment is to evaluate the effectiveness of the discrete transform technique for solving the Newton equations, we only report the time per iteration, and omit the number of iterations or the accuracy reached. Some columns are incomplete due to the 'out of memory' error caused while solving the larger problems. Note that we use odd values of $n_1 = n_2$ for the first example and even values for the second. This was done in order to verify that the parity description (presented in [10] and affecting the choice of trigonometric polynomial basis functions) works in both cases. From the table we observe that the time per iteration for the general-purpose solvers grows at a rate between $O(n^6)$ and $O(n^7)$.

Table 2 shows the solution times per iteration for the more complex diamond filter problems. We note that the SeDuMi times appear to increase at a slower rate (but still greater than $O(n^6)$), indicating that the exploitation of sparsity is quite effective for this family of

split the free variable term as a difference form $By^+ - By^-$, with nonnegative vector variables y^+ and y^-. However, this is known to cause numerical difficulties. Secondly, SDPT3 v. 4.0 supports only a single semidefinite variable block, while our experiments require multiple blocks. We therefore did not use the low-rank option in SDPT3. We point out that the inclusion of free variables in DSDP, or multiple diagonal blocks in SDPT3, would make these packages as well suited for the SDPs described here as our customized Matlab solver.

$n = n_1 = n_2$	Gram-pair			DT-based
	SeDuMi	DSDP	SDPT3	IPM
5	0.07	0.09	0.23	0.10
7	0.21	0.32	0.95	0.37
9	1.03	0.99	2.15	1.15
11	3.15	3.34	4.58	2.97
13	9.16	7.36	9.90	6.78
15	24.4	17.1	19.8	14.1
17	49.1	43.8	40.5	26.2
19		86.9		47.2
21				80.6
23				132
25				212

Table 1: Solution times per iteration (in seconds) for the lowpass filter design problem.

$n = n_1 = n_2$	Gram-pair			DT-based
	SeDuMi	DSDP	SDPT3	IPM
6	0.44	0.465	0.966	0.32
8	2.30	1.55	5.29	0.93
10	8.41	5.36	12.0	4.24
12	21.7	13.2	25.63	7.10
14	52.1	33.3		14.3
16				31.8
18				68.6
20				129

Table 2: Solution times per iteration (in seconds) for the diamond filter design problem.

problems. Another observation is that the complexity growth for DSDP is between $O(n^5)$ and $O(n^6)$. This can be explained by the fact that DSDP takes advantage of low-rank structure.

In both tables the times for the fast discrete-transform based method, within this relatively small range of values for n, appear to grow at a rate of about $O(n^5)$.

6 Conclusion

We have discussed a fast implementation of primal-dual interior-point algorithms for multivariate trigonometric sum-of-squares programs, based on the sampling formulation proposed in [17, 24]. This approach leads to SDPs with a low-rank structure, which is easily exploited in standard interior-point algorithms. For two-dimensional FIR filter design problems with filters of order n, the fast algorithm has a complexity of $O(n^6)$ per iteration. Experimental results confirm that this method is more efficient and requires less memory than general-purpose solvers applied to the standard sparse Gram matrix representation.

References

[1] F. Alizadeh, J.-P. A. Haeberly, and M. L. Overton, "Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results," *SIAM J. on Optimization*, vol. 8, no. 3, pp. 746–768, 1998.

[2] B. Alkire and L. Vandenberghe, "Convex optimization problems involving finite autocorrelation sequences," *Mathematical Programming Series A*, vol. 93, pp. 331–359, 2002.

[3] S. J. Benson and Y. Ye and X. Zhang. "Solving Large-Scale Sparse Semidefinite Programs for Combinatorial Optimization," *SIAM J. on Optimization*, vol. 10, no. 2, pp. 443–461, 2000.

[4] S. J. Benson and Y. Ye. "DSDP5: Software for semidefinite programming," Technical report ANL/MCS-P1289-0905, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL. Submitted to ACM Transactions on Mathematical Software, September, 2005. `http://www.mcs.anl.gov/~benson/dsdp`.

[5] S. J. Benson and Y. Ye. "DSDP5 User Guide — Software for semidefinite programming," Technical report ANL/MCS-TM-227, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL. Submitted to ACM Transactions on Mathematical Software, September, 2005. `http://www.mcs.anl.gov/~benson/dsdp`.

[6] W. L. Briggs and V. E. Henson. *The DFT. An Owner's Manual for the Discrete Fourier Transform.* SIAM, 1995.

[7] T. N. Davidson, Z.-Q. Luo, and J. F. Sturm. Linear matrix inequality formulation of spectral mask constraints. *IEEE Transactions on Signal Processing*, 50(11):2702–2715, 2002.

[8] M.A. Dritschel, "On Factorization of Trigonometric Polynomials," *Integr. Equ. Oper. Theory*, vol. 49, pp. 11–42, 2004.

[9] B. Dumitrescu, "Trigonometric polynomials positive on frequency domains and applications to 2-D FIR filter design," *IEEE Trans. Signal Proc.*, vol.54, no.11, pp. 4282–4292, Nov. 2006.

[10] B. Dumitrescu, "Gram pair parameterization of multivariate sum-of-squares trigonometric polynomials," *EUSIPCO*, Florence, Italy, Sept. 2006.

[11] B. Dumitrescu. *Positive Trigonometric Polynomials and Signal Processing Applications*. Spinger, 2007.

[12] M. Fukuda, M. Kojima, K. Murota, and K. Nakata. "Exploiting sparsity in semidefinite programming via matrix completion I: general framework," *SIAM Journal on Optimization*, vol. 11, pp. 647–674, 2000.

[13] Y. Genin, Y. Hachez, Yu. Nesterov, and P. Van Dooren. Optimization problems over positive pseudopolynomial matrices. *SIAM Journal on Matrix Analysis and Applications*, 25(1):57–79, 2003.

[14] C. Helmberg, F. Rendl, R. Vanderbei, and H. Wolkowicz. "An interior-point method for semidefinite programming," *SIAM J. on Optimization*, vol. 6, pp. 342–361, 1996.

[15] M. Kojima, S. Shindoh, S. Hara. "Interior-point methods for the monotone linear complementarity problem in symmetric matrices," *SIAM J. on Optimization*, vol. 7, pp. 86–125, 1997.

[16] J. B. Lasserre, "Global optimization with polynomials and the problem of moments," *SIAM Journal on Optimization*, vol. 11, no. 3, pp. 796–817, 2001.

[17] J. Löfberg and P. A. Parrilo, "From coefficients to samples: a new approach to SOS optimization," in *Proceedings of the 43rd IEEE Conference on Decision and Control*, 2004, pp. 3154–3159.

[18] A. Megretski, "Positivity of trigonometric polynomials," in *Proceedings of the 42nd IEEE Conference on Decision and Control*, 2003, pp. 3814–3817.

[19] K. Nakata, K. Fujitsawa, M. Fukuda, M. Kojima, and K. Murota. "Exploiting sparsity in semidefinite programming via matrix completion II: implementation and numerical details," *Mathematical Programming Series B*, vol. 95, pp. 303–327, 2003.

[20] Y. Nesterov, "Squared functional systems and optimization problems," in *High Performance Optimization Techniques*, J. Frenk, C. Roos, T. Terlaky, and S. Zhang, Eds., pp. 405–440. Kluwer Academic Publishers, 2000.

[21] P. A. Parrilo, "Semidefinite programming relaxations for semialgebraic problems," *Mathematical Programming Series B*, vol. 96, pp. 293–320, 2003.

[22] N.Z. Shor, "Class of Global Minimum Bounds of Polynomial Functions," *Cybernetics*, vol. 23, no. 6, pp. 731–734, 1987, (Russian orig.: Kibernetika, no.6, pp.9–11, 1987).

[23] J.W. McLean and H.J. Woerdeman, "Spectral Factorizations and Sums of Squares Representations via Semidefinite Programming," *SIAM J. Matrix Anal. Appl.*, vol. 23, no. 3, pp. 646–655, 2002.

[24] T. Roh and L. Vandenberghe, "Discrete transforms, semidefinite programming and sum-of-squares representations of nonnegative polynomials," *SIAM J. on Optimization*, vol. 16, no. 4, pp. 939–964, 2006.

[25] T. Roh, B. Dumitrescu, and L. Vandenberghe. "Interior-point algorithms for sum-of-squares optimization of multidimensional trigonometric polynomials," In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 905–908, April 15–20, 2007, Honolulu, Hawai'i.

[26] J.F. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optimization Methods and Software*, vol. 11–12, pp. 625–653, 1999, Version 1.1 available from `http://sedumi.mcmaster.ca`.

[27] M. J. Todd, K. C. Toh, and R. H. Tütüncü, "On the Nesterov-Todd direction in semidefinite programming," *SIAM J. on Optimization*, vol. 8, no. 3, pp. 769–796, 1998.

[28] K. C. Toh, R. H. Tütüncü and M. J. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical Programming Series B*, 95:189–217, 2003.

[29] M. J. Todd, K. C. Toh, and R. H. Tütüncü, "On the implementation and usage of SDPT3: a Matlab software package for semidefinite-quadratic-linear programming, version 4.0." Available from `http://www.math.nus.edu.sg/~mattohkc/sdpt3.html`. July, 2006.

[30] P. P. Vaidyanathan. *Multirate Systems and Filter Banks*. Prentice Hall, 1993.

[31] H. Waki, S. Kim, M. Kojima, and M. Muramatsu, "Sums of squares and semidefinite program relaxations for polynomial optimization problems with structured sparsity," *SIAM J. on Optimization*, vol. 17, no. 1, pp. 218–242, 2006.

[32] S.-P. Wu and S. Boyd and L. Vandenberghe. "FIR Filter Design via Spectral Factorization and Convex Optimization," In B. Datta, editor, *Applied and Computational Control, Signals, and Circuits*, volume 1, pages 215–245. Birkhauser, 1998.