FIR FILTER OPTIMIZATION WITH POS3POLY IN CVX

Bogdan C. Șicleru¹, Bogdan Dumitrescu¹ ²

¹Dept. of Automatic Control and Computers
"Politehnica" University of Bucharest
313 Spl. Independenței, 060042 Bucharest, Romania
e-mail: bogdan.sicleru@schur.pub.ro, bogdan.dumitrescu@tut.fi

²Tampere Int. Center for Signal Processing
Tampere University of Technology
P.O. Box 553, SF-33101 Tampere, Finland

ABSTRACT

POS3POLY is a new library for solving convex optimization problems with positive polynomials, designed and written by the current authors. POS3POLY allows simple descriptions of three types of positive multivariate polynomials, namely trigonometric, real and hybrid, sparing the user of the task of parameterizing them with linear matrix inequalities. Positivity on semi-algebraic domains and a polynomial Bounded Real Lemma are also treated. In this paper, we present the features of POS3POLY that allow the manipulation of positive trigonometric polynomials in CVX and are useful in FIR filter optimization. Three fully detailed examples are given, for designing peak-constrained least-squares linear-phase filters, sparse 2D linear-phase filters and 2D approximately linear-phase filters.

1. INTRODUCTION

FIR filter design is one of the main beneficiaries of the developments in convex optimization [3]. Many recent methods for optimizing one- or multi-dimensional FIR filters appeal to linear, quadratic and semidefinite programming (SQLP) as a basic tool. The design is facilitated by several software libraries appeared in the last decade as a practical outcome of the theoretical progress of convex optimization methods. For SQLP, leading examples are SeDuMi [11] and SDPT3 [12], in which the SQLP problem has to be described in an elementary standard form. A further step in easing designer’s task was taken with CVX [6], which allows modeling convex problems, including but not limited to SQLP, using a natural mathematical description of an optimization problem; the transformation to an equivalent SQLP language similar to the mathematical description of an optimization problem and the more optimization-knowledgeable user, who is familiar with SeDuMi but does not have to bother with the intricacies of positive polynomials parameterizations.

POS3POLY is a new library for solving convex optimization problems with positive polynomials. POS3POLY solves the problem

\[
\begin{aligned}
\min & \ c^T x \\
\text{s.t.} & \ Ax = b, & x \in K \times \mathcal{P}
\end{aligned}
\]

where \(K\) is the SQLP cone (cartesian product of nonnegative orthant, second order cone and semidefinite matrices cone) and \(\mathcal{P}\) is a cartesian product of diverse cones of positive polynomials. POS3POLY transforms (1) into the standard SQLP problem

\[
\begin{aligned}
\min & \ c^T x \\
\text{s.t.} & \ Ax = b, & x \in K
\end{aligned}
\]

by taking advantage of the available parameterizations of sum-of-squares polynomials via semidefinite (Gram) matrices. For multivariate polynomials, (2) is a sum-of-squares relaxation of the original problem (1), hence may not have the same solution. However, as revealed by empirical evidence, the solution of (2) in practical problems is usually identical or very close to the solution of (1).

POS3POLY can also be used for creating the set of positive polynomials inside CVX. In CVX, the linear constraints need not be expressed as a single system \(Ax = b\) and equalities can be mixed with inequalities. However, after internal transformations, CVX solves a problem of form (2). Hence, POS3POLY offers support for both the high-end user who needs only recognize and summarily describe the convex optimization problem and the more optimization-knowledgeable user, who is familiar with SeDuMi but does not have to bother with the intricacies of positive polynomials parameterizations.

POS3POLY allows working with three types of polynomials (hence the ‘3’ in the name): trigonometric, real and hybrid (real-trigonometric). The polynomials can be univariate or multivariate, having real or complex, scalar or matrix coefficients. The positivity can be global or only on semi-algebraic domains (intervals in the univariate case). A Bounded Real Lemma (BRL) for polynomials is also implemented. In this paper, we present the features of POS3POLY that allow the manipulation of positive trigonometric polynomials in CVX and are useful in FIR filter optimization. Three fully detailed examples are given, for designing peak-constrained least-squares linear-phase filters, sparse 2D linear-phase filters and 2D approximately linear-phase filters.

FIR FIL TER OPTIMIZATION WITH POS3POL Y IN CVX

Bogdan C. Șicleru¹, Bogdan Dumitrescu¹ ²

¹Dept. of Automatic Control and Computers
"Politehnica" University of Bucharest
313 Spl. Independenței, 060042 Bucharest, Romania
e-mail: bogdan.sicleru@schur.pub.ro, bogdan.dumitrescu@tut.fi

© EURASIP, 2011 - ISSN 2076-1465

This work was supported by CNCSIS-UEFISCSU, project PNII – IDEI 309/2007, Teke’s FiDiPro – Finland Distinguished Professor Programme and the Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family and Social Protection through the Financial Agreement POSDRU/6/1.5/S/16.

1. INTRODUCTION

FIR filter design is one of the main beneficiaries of the developments in convex optimization [3]. Many recent methods for optimizing one- or multi-dimensional FIR filters appeal to linear, quadratic and semidefinite programming (SQLP) as a basic tool. The design is facilitated by several software libraries appeared in the last decade as a practical outcome of the theoretical progress of convex optimization methods. For SQLP, leading examples are SeDuMi [11] and SDPT3 [12], in which the SQLP problem has to be described in an elementary standard form. A further step in easing designer’s task was taken with CVX [6], which allows modeling convex problems, including but not limited to SQLP, using a natural mathematical description of an optimization problem; the transformation to an equivalent SQLP language similar to the mathematical description of an optimization problem and the more optimization-knowledgeable user, who is familiar with SeDuMi but does not have to bother with the intricacies of positive polynomials parameterizations.

POS3POLY is a new library for solving convex optimization problems with positive polynomials. POS3POLY solves the problem

\[
\begin{aligned}
\min & \ c^T x \\
\text{s.t.} & \ Ax = b, & x \in K \times \mathcal{P}
\end{aligned}
\]

where \(K\) is the SQLP cone (cartesian product of nonnegative orthant, second order cone and semidefinite matrices cone) and \(\mathcal{P}\) is a cartesian product of diverse cones of positive polynomials. POS3POLY transforms (1) into the standard SQLP problem

\[
\begin{aligned}
\min & \ c^T x \\
\text{s.t.} & \ Ax = b, & x \in K
\end{aligned}
\]

by taking advantage of the available parameterizations of sum-of-squares polynomials via semidefinite (Gram) matrices. For multivariate polynomials, (2) is a sum-of-squares relaxation of the original problem (1), hence may not have the same solution. However, as revealed by empirical evidence, the solution of (2) in practical problems is usually identical or very close to the solution of (1).

POS3POLY can also be used for creating the set of positive polynomials inside CVX. In CVX, the linear constraints need not be expressed as a single system \(Ax = b\) and equalities can be mixed with inequalities. However, after internal transformations, CVX solves a problem of form (2). Hence, POS3POLY offers support for both the high-end user who needs only recognize and summarily describe the convex optimization problem and the more optimization-knowledgeable user, who is familiar with SeDuMi but does not have to bother with the intricacies of positive polynomials parameterizations.

POS3POLY allows working with three types of polynomials (hence the ‘3’ in the name): trigonometric, real and hybrid (real-trigonometric). The polynomials can be univariate or multivariate, having real or complex, scalar or matrix coefficients. The positivity can be global or only on semi-algebraic domains (intervals in the univariate case). A Bounded Real Lemma (BRL) for polynomials is also implemented. In this paper, we present the features of POS3POLY that allow the manipulation of positive trigonometric polynomials in CVX and are useful in FIR filter optimization. Three fully detailed examples are given, for designing peak-constrained least-squares linear-phase filters, sparse 2D linear-phase filters and 2D approximately linear-phase filters.
2. POSITIVE TRIGONOMETRIC POLYNOMIALS

A symmetric (zero-phase) d-variate trigonometric polynomial has the form

$$R(z) = \sum_{k=-n}^{n} r_k z^{-k}, \quad r_k = r_k^*, \quad (3)$$

where $z = (z_1, z_2, \ldots, z_d)$, $z^k = z_1^{k_1} \cdots z_d^{k_d}$ and $^*$ denotes complex conjugation; the sum in (3) goes for all $k \in \mathbb{N}^d$ such that $-n \leq k \leq n$. The polynomial has real values on the unit d-circle $\mathbb{T}^d$. Due to symmetry, the coefficients defining the polynomial belong to a halfspace; for convenience, we use the halfspace $H_d \subseteq \mathbb{Z}^d$ defined by: $k \in H_d$ if $(k_1) > 0$ or $(k_d = 0$ and $(k_1, \ldots, k_{d-1}) \in H_{d-1})$. In POS3POLY, the polynomial is described by the vector

$$[r_0, r_1, \ldots, r_n, r_n, \ldots, r_{-1}]^T.$$ (4)

The length of this vector is $[1 + \prod_{i=1}^d(2n_i + 1)]/2$.

For example, the 1D polynomial (or linear-phase FIR filter)

$$R(z) = 3 + 2(z^{-1} + z) + (z^{-2} + z^2)$$

is represented through the vector

$$r = [3 2 1]^T;$$

containing the coefficients of its causal part.

The 2D polynomial of degree $(1, 1)$

$$R(z_1, z_2) = 5 + 4(z_1^{-1} + z_2^{-1}) + 3(z_1^{-2}z_2 + z_1^{-1}z_2^{-1}) + 2(z_1^{-2} + z_2^{-2}) + (z_1^{-1}z_2^{-1} + z_1z_2)$$

is represented through the 5-elements vector

$$r = [5 4 3 2 1]^T;$$

As the simplest example of POS3POLY use, let us compute the minimum value of the above 2D polynomial on the unit bicircle. This is equivalent to finding the largest $\mu$ such that $R(z) - \mu \geq 0$. The corresponding CVX program is

```cvx_begin
variable mu;
maximize mu
subject to
  r - mu*eye(5,1) == sos_poly([1 1 1]);
 cvx_end```

The function sos_poly creates variables that are positive polynomials. (By default, they are trigonometric polynomials.) The first argument of sos_poly has the form $[n, k]$, i.e. contains the degree of the polynomial and the size of its coefficients ($k = 1$ for scalar coefficients). If only this argument is given, then the polynomial is globally positive. Since POS3POLY uses sum-of-squares relaxation, the polynomial is actually sum-of-squares of degree $n$. Running the above program gives $\mu = -7$, which is the true minimum value.

In the case of matrix coefficients, where $R_k \in \mathbb{C}^{k \times k}$ replaces $r_k$ in (3), the symmetry relation writes $R_{-k} = R_k^H$ (where the index $H$ means transposition and complex conjugation). The polynomial is represented by a vector in the style of (4), with $r_k$ replaced by vec$(R_k)$ (for $k \neq 0$) where the operator vec concatenates the columns of $R_k$. Since $R_0$ is Hermitian, it is described by vectorizing (on columns) its lower triangular part.

3. POSITIVITY ON INTERVALS

More generally, POS3POLY handles polynomials that are positive on certain sets. In this section we present the case of 1D polynomials, where one can describe positivity on intervals or unions of disjoint intervals. Let such a union be

$$\mathcal{I} = \bigcup_{\ell=1}^{L}[\omega_{1\ell}, \omega_{2\ell}]. \quad (5)$$

To describe a polynomial that is positive on $\mathcal{I}$ (and other possible properties), the function sos_poly needs a second argument, call it ptype, which is a structure. In this case, only one field of the structure is used, namely ptype.int, having the value

$$[\omega_{11} \omega_{21} \omega_{12} \omega_{22} \ldots \omega_{1L} \omega_{2L}].$$

(We will see later the use of other fields of ptype.)

As an introductory filter design example, let us consider the peak-constrained least-square (PCLS) optimization [1] of the linear phase filter of order $2n$ given by

$$z^{-n}H(z) = z^{-n} \sum_{k=-n}^{n} h_k z^{-k}, \quad h_{-k} = h_k.$$ Assumed for simplicity that a lowpass filter is desired, the passband and stopband edges $\omega_p$ and $\omega_s$ are given, together with the passband and stopband error bounds $\gamma_p$ and $\gamma_s$, respectively. The PCLS problem consists of minimizing the stopband energy $E_s$ subject to peak constraints in passband and stopband:

$$\min E_s \quad \text{s.t.} \quad [1 - H(e^{i\omega})] \leq \gamma_p, \quad \forall \omega \in [0, \omega_p]$$

$$[H(e^{i\omega})] \leq \gamma_s, \quad \forall \omega \in [\omega_s, \pi]$$

Denoting $h = [h_0, h_1 \ldots h_n]^T$, the stopband energy of the filter is

$$E_s = h^T \mathbf{C} h, \quad \text{with} \quad \mathbf{C} = \mathbf{P}^T \mathbf{C} \mathbf{P} \succeq 0; \quad (6)$$

where

$$\mathbf{P} = \begin{bmatrix} 0 & J_n \\
 1 & 0 \\
 0 & I_n \end{bmatrix}$$

($J_n$ is the counteridentity matrix of size $n \times n$) and $\mathbf{C} = \text{Toeplitz}(c_0, c_1, \ldots, c_n) \succeq 0$, with

$$c_k = \begin{cases} -\omega_p / \pi, \quad & \text{if } k = 0; \\
 -\sin k\omega_s / k\pi, \quad & \text{if } k > 0. \end{cases}$$

1406
By showing explicitly the positive polynomials, extending to upper passband bound to the whole frequency domain and expressing the minimization of stopband energy as a second-order cone constraint, the problem (6) becomes

\[
\begin{align*}
\min & \quad \epsilon \\
\text{s.t.} & \quad |C^T h| \leq \epsilon \\
& \quad 1 + \gamma_p^2 H(e^{j\omega}) \geq 0, \quad \forall \omega \\
& \quad H(e^{j\omega}) - 1 + \gamma_p \geq 0, \quad \forall \omega \in [0, \omega_p] \\
& \quad \gamma_p - H(e^{j\omega}) \geq 0, \quad \forall \omega \in [\omega_s, \pi] \\
& \quad H(e^{j\omega}) + \gamma_s \geq 0, \quad \forall \omega \in [0, \omega_p] 
\end{align*}
\]

Let us design a PCLS filter with \( n = 34, \omega_p = 0.3\pi, \omega_s = 0.36\pi, \gamma_p = \gamma_s = 0.01 \). We start by introducing the data:

\[
\begin{align*}
n &= 34; \\
wp &= 0.3\pi; \\
ws &= 0.36\pi; \\
gp &= 0.01; \\
gs &= 0.01; 
\end{align*}
\]

Then, we build the matrix describing the stopband energy criterion

\[
N = 2n + 1;
\]

\[
c = \text{zeros}(N, 1);
\]

\[
c(1) = 1 - ws / \pi; \\
c(2:N) = -\sin((1:N-1)*ws) ./ (1:N-1)/pi; \\
P = [zeros(n,1) fliplr(eye(n)); ... 1 zeros(1,n); ... zeros(n,1) eye(n)]; \\
C = \text{real}(\text{sqrtm}(P' * \text{toeplitz}(c) * P));
\]

Next, we define the structures that characterize positivity in passband and stopband

\[
\begin{align*}
\text{ptype}_{\epsilon p}.\text{int} &= [0 \ wp]; \\
\text{ptype}_{\epsilon s}.\text{int} &= [ws \ wp];
\end{align*}
\]

We are now ready to describe problem (8), which is simply

\[
\begin{align*}
\text{pl} &= \text{eye}(n+1,1); \\
\text{cvx}_\text{begin} \\
\text{variable} &\quad e; \\
\text{variable} &\quad h(n+1); \\
\text{minimize} &\quad e; \\
\text{subject to} &\quad \text{norm}(C * h) <= e; \\
&\quad (1+gp) * \text{pl} - h == \text{sos}_\text{pol}(\{n 1\}); \\
&\quad h - (1-gp) * \text{pl} == \text{sos}_\text{pol}(\{n 1\}, \text{ptype}_{\epsilon s}); \\
&\quad gs * \text{pl} - h == \text{sos}_\text{pol}(\{n 1\}, \text{ptype}_{\epsilon p}); \\
&\quad h + gs * \text{pl} == \text{sos}_\text{pol}(\{n 1\}, \text{ptype}_{\epsilon p}); \\
\text{cvx}_\text{end}
\end{align*}
\]

The code is practically self-explanatory! There is no need to know anything about the parameterization of trigonometric polynomials that are positive on an interval. Finally, we compose the full vector of coefficients of the linear-phase filter by

\[
H = [h(\text{end}:-1:1); \ h(2:\text{end})];
\]

The frequency response of the designed filter is shown in Figure 1.
can be described by polynomial positivity through
\[
\mathcal{P}_p = \{ z \in \mathbb{T}^2 : D_{p_1}(z) \geq 0, D_{p_2}(z) \geq 0, D_{p_3}(z) \geq 0 \}
\]
\[
\mathcal{P}_s = \mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{P}_3
\]
\[
\mathcal{P}_1 = \{ z \in \mathbb{T}^2 : D_{p_1}(z) \geq 0 \}
\]
\[
\mathcal{P}_2 = \{ z \in \mathbb{T}^2 : D_{p_2}(z) \geq 0 \}
\]
\[
\mathcal{P}_3 = \{ z \in \mathbb{T}^2 : D_{p_3}(z) \geq 0 \}
\]
Denoting \( d_p = \cos \theta_p, a_n = \cos \omega_n \), the polynomials appearing above are
\[
D_{p_1}(z_1,z_2) = (\cos \theta_1 + \cos \theta_2 2 - a_p
\]
\[
D_{p_2}(z_1,z_2) = (\cos \theta_1 + \cos \theta_2 2 - a_p
\]
\[
D_{p_3}(z_1,z_2) = (\cos \theta_1 + \cos \theta_2 2 + (z_2 + z_2^2)/2
\]
\[
D_{s_1}(z_1,z_2) = -(\cos \theta_1 + \cos \theta_2 2 + a_s
\]
\[
D_{s_2}(z_1,z_2) = -(\cos \theta_1 + \cos \theta_2 2 + a_s
\]
\[
D_{s_3}(z_1,z_2) = -(\cos \theta_1 + \cos \theta_2 2 - (z_2 + z_2^2)/2
\]
\begin{equation}
\tag{11}
\end{equation}
For a polynomial that is positive on \( \mathcal{P}_p \), the field \( \text{ptype} \) is described for POS3POLY via the following lines (ap stands for \( d_p \))
\[
\text{ptypep.dom.deg} = [1 1; 1 1; 1 1];
\]
\[
\text{ptypep.dom.coef} = [-ap 0 0 0 0.5 ... ap 0 0.5 0 0 ... 0 0.5 0 0.5 0];
\]
Since there is no field \( \text{nunion} \), it is assumed that \( L = 1 \) in (9). The field \( \text{deg} \) says that there are \( I_1 = 3 \) polynomials whose positivity characterizes \( \mathcal{P}_p \), all of degree (1,1). The field \( \text{coef} \) contains the vector of concatenated coefficients of the first three polynomials from (11).
Similarly, for \( \mathcal{P}_s \), the description is
\[
\text{ptypes.dom.nunion} = [1 1 1];
\]
\[
\text{ptypes.dom.deg} = [1 1; 1 1; 1 1];
\]
\[
\text{ptypes.dom.coef} = [as 0 0 0 -0.5 ... as 0 -0.5 0 0 ... 0 -0.5 0 0.5 0];
\]
Now, the field \( \text{nunion} \) tells that \( L = 3 \) and \( I_1 = I_2 = I_3 = 1 \) in (9). The fields \( \text{deg} \) and \( \text{coef} \) are filled using the degrees and coefficients of the last three polynomials from (11).
Assume that we want to design sparse 2D filters as in [9] (see an alternative approach in the 1D case in [2]), starting from the solution of the minimization problem
\[
\begin{align*}
\text{min} & & \frac{\gamma + \lambda}{\|r\|_1} \\
\text{s.t.} & & R(z) \leq 1 + \gamma, \quad \forall z \in \mathbb{T}^2 \\
& & R(z) \geq 1 - \gamma, \quad \forall z \in \mathcal{P}_p \\
& & R(z) \leq \gamma, \quad \forall z \in \mathcal{P}_s \\
& & R(z) \geq -\gamma, \quad \forall z \in \mathcal{P}_s
\end{align*}
\tag{12}
\]
The optimization criterion mixes the maximum error bound \( \gamma \) (with respect to the ideal response) and the 1-norm of the vector of coefficients of the filter, with the aim of promoting sparsity. The positive constant \( \lambda \) is a trade-off weight. Assume that we want to design a filter (3) with \( n = (8,8) \) (having a total of 289 coefficients, of which 145 distinct, this being the length of the vector \( r \)), taking \( \omega_p = 0.5\pi, \omega_s = 0.9\pi, \lambda = 0.001 \). We set first the design data
\[
n = [8 8];
\]
\[
ap = \cos(0.5*\pi); \quad as = \cos(0.9*\pi);
\]
\[
\text{lam} = 0.001;
\]
Figure 2: Frequency response of designed sparse 2-D filter.

Then, we set \( \text{ptypep} \) and \( \text{ptypes} \) as described above. For solving (12), we use the following sequence of instructions
\[
\begin{align*}
M &= (1 + \text{prod}(2*n+1))/2; \\
p1 &= \text{eye}(M,1); \\
cvx_begin \\
\text{variable g; } \\
\text{variable r(M); } \\
\text{minimize g + lam*norm(r(1,1)); } \\
\text{subject to } \\
(1+g)*p1 - r &= \text{sos_pol([n 1]); } \\
r - (1-g)*p1 &= \text{sos_pol([n 1, ptypep]; } \\
g*p1 - r &= \text{sos_pol([n 1, ptypes]); } \\
r + g*p1 &= \text{sos_pol([n 1, ptypes]); } \\
cvx_end
\end{align*}
\]
Notice the similarity (in form) with the 1D problem from the previous section. The sum-of-squares parameterization [4] that replaces (and relaxes) the polynomial inequalities from (12) is hidden.

The solution of (12) can be used for further optimization. Assume that we want a filter with only \( N \) nonzero elements.
We select the \( N \) largest coefficients of the solution of (12) and solve again (12) with \( \lambda = 0 \) and forcing to zero the other coefficients; the above code can be easily modified to this purpose, so we skip the programming details. Hence, the optimal values of the selected coefficients are obtained.
For example, by taking \( N = 193 \) (remind that the full filter has 289 coefficients), the optimal error is \( \gamma = 0.00559 \). The frequency response is shown in Figure 2. By reoptimizing using the largest \( N \) coefficients of the optimal full filter (obtained by solving (12) with \( \lambda = 0 \)), we get \( \gamma = 0.00672 \). So, it indeed makes sense to introduce the 1-norm term in the criterion, as shown also in [9], where all optimization is based on discretization and linear programming.

5. BOUNDED REAL LEMMA

A causal trigonometric polynomial with matrix coefficients has the form
\[
H(z) = \sum_{k=0}^{n} H_k z^{-k}, \tag{13}
\]
with $H_k \in \mathbb{C}^{k^2 \times k^2}$. In POS3POLY, it is described by the vector
\[
\begin{align*}
& \text{vec}(H_0_{-0}); \text{vec}(H_{1,0_{-0}}); \ldots; \text{vec}(H_{n_1,0_{-0}}); \\
& \text{vec}(H_{0,1_{-0}}); \ldots; \text{vec}(H_{n_1,1_{-n_2}}) \big] \big) \big] \big].
\end{align*}
\]

A Bounded Real Lemma (BRL) is an inequality
\[
\|H(z)\|_{\infty} \leq |A(z)|, \quad \forall z \in \mathcal{D},
\]
where $A(z)$ is a causal polynomial with degree $n$, but with scalar coefficients, and $\mathcal{D}$ is defined as in (9). Such a BRL can be described in POS3POLY through a single vector variable that concatenates the vectors $(14)$ for $H(z)$ and $(4)$ for the symmetric polynomial $R(z) = A(z)A(z^{-1})$. That such a variable corresponds to a BRL is marked by the presence of the field $\text{ptypep.brl}$. The only subfield needed for working with CVX is $\text{hsize}$, which should contain the vector $[\kappa \kappa]$ giving the size of the matrix coefficients.

For illustration, let us consider the minimax design of approximately linear phase filters via the optimization problem
\[
\begin{align*}
\min_{\gamma} \quad & \gamma \\
\text{s.t.} \quad & |H(z)| - \gamma z^{n} \\ & |H(z)| \leq \gamma, \quad \forall z \in \mathcal{D}
\end{align*}
\]
where the passband and stopband domains are defined as in (10) and $\tau_1, \tau_2 \in \mathbb{N}$. To solve this problem with POS3POLY, we first set $n, \tau_1, \tau_2$ to the desired values of $n$, $\tau_1$, $\tau_2$, respectively. We define $\text{ptypep}$ and $\text{ptypes}$ like in the previous section for describing the diamond-shaped domains and add
\[
\begin{align*}
\text{ptypep.brl.hsize} &= [1 1]; \\
\text{ptypes.brl.hsize} &= [1 1];
\end{align*}
\]
for specifying that we deal with BRL inequalities. Next, we define a few auxiliary variables:
\[
\begin{align*}
M_h &= \text{prod}(n+1); \\
M_r &= (1 + \text{prod}(2*n+1)) / 2; \\
t &= \text{zeros}(M_h+M_r,1); \\
t(\tau_1*2*(n+1)+\tau_2+1 ) &= 1;
\end{align*}
\]
\[M_h\] and $M_r$ are the lengths of the vectors of coefficients of $H(z)$ and $R(z) = \gamma^2$; note that although in this case $R(z)$ is a constant, the corresponding coefficient vector is as long as for a polynomial of degree $n$. The vector $t$ represents the monomial $z_1^{\tau_1}z_2^{\tau_2}$. Denoting $\gamma^2$ by $g$, the problem (16) is solved by the code
\[
\begin{align*}
\text{cvx_begin} \\
\text{variable} g; \\
\text{variable} hr(M_h+M_r); \\
\text{minimize} g; \\
\text{subject to} \\
hr - t &= \text{sos pol}([n 1], \text{ptypep}); \\
hr &= \text{sos pol}([n 1], \text{ptypes}); \\
hr(M_h+1:end) &= g * \text{eye}(M_r,1); \\
\text{cvx_end}
\end{align*}
\]
The first two equality constraints correspond to the BRL inequalities from (16), while the third represents the equality $R(z) = \gamma^2$. 

6. CONCLUSIONS

The library POS3POLY lightens the programming burden in convex optimization with positive polynomials, as we hope that this paper made clear. The interested reader should go to www.schur.pub.ro/pos3poly for more information and the software itself. The user’s guide contains many design examples, including the SeDuMi version of the FIR filter design problems treated here (also combining minimax and least-squares optimization), but also regarding the design of matrix filters, adjustable FIR filters, MIMO filters. Further work will be directed to implementing faster alternative parameterizations.

REFERENCES