gKYPSDP User Guide

Zhang Liu and Lieven Vandenberghe*

1 Introduction

gKYPSDP is a Matlab program for solving semidefinite programs (SDPs) derived from the Kalman-Yakubovich-Popov (KYP) lemma and its generalizations. These SDPs have the following structure:

$$\begin{array}{ll} \text{minimize} & w^T x \\ \text{subject to} & \begin{bmatrix} A_i & B_i \\ I & 0 \end{bmatrix}^T (\Phi_i \otimes P_i + \Psi_i \otimes Q_i) \begin{bmatrix} A_i & B_i \\ I & 0 \end{bmatrix} + M_i(x) + N_i \preceq 0, \\ & Q_i \succeq 0, \quad i = 1, 2, \dots, L. \end{array} \tag{1}$$

Here, the problem data are $w \in \mathbf{R}^p$, $A_i \in \mathbf{R}^{n_i \times n_i}$, $B_i \in \mathbf{R}^{n_i \times m_i}$, $\Phi_i \in \mathbf{H}^2$, $\Psi_i \in \mathbf{H}^2$, $N_i \in \mathbf{H}^{n_i+m_i}$, and the mapping $M(x) = \sum_{j=1}^p x_j M_{ij}$ with $M_{ij} \in \mathbf{H}^{n_i+m_i}$. The optimization variables are $x \in \mathbf{R}^p$, $P_i \in \mathbf{H}_i^n$, and $Q_i \in \mathbf{H}^{n_i}$. $\otimes$ is the Kronecker product. Problems of this form are widely encountered in control and signal processing.

gKYPSDP is based on a custom primal-dual interior-point method, which exploits the structure in the constraints to achieve a higher efficiency than general-purpose SDP solvers. The method is particularly fast when the number of inputs, m_i, is small. Details of the algorithm and references to applications can be found in [LV07].

2 The Kalman-Yakubovich-Popov lemma

2.1 The KYP lemma

Suppose $\Phi \in \mathbf{H}^2$ is a matrix with negative determinant, and (A, B) is a controllable pair. We define

$$U(\lambda) = (\lambda I - A)^{-1} B, \qquad V(\lambda) = \begin{bmatrix} U(\lambda) \\ I \end{bmatrix}.$$

*Electrical Engineering Department, University of California, Los Angeles. Email: zhang@ee.ucla.edu, vandenbe@ee.ucla.edu. Research supported by NSF under grant ECS-0524663.

The KYP lemma states that the frequency-domain inequality (FDI)

$$F(\lambda) = V(\lambda)^H M V(\lambda) \preceq 0 \tag{2}$$

holds for all $\lambda \in \mathbf{C}$ that satisfy

$$\begin{bmatrix} \lambda \\ 1 \end{bmatrix}^H \Phi \begin{bmatrix} \lambda \\ 1 \end{bmatrix} = 0$$

if and only if there exists a $P \in \mathbf{H}^n$ such that

$$\begin{bmatrix} A & B \\ I & 0 \end{bmatrix} (\Phi \otimes P) \begin{bmatrix} A & B \\ I & 0 \end{bmatrix} + M \preceq 0. \tag{3}$$

For a continuous-time system, we are interested in a frequency domain condition on the imaginary axis, *i.e.*, $\lambda = j\omega$ and

$$\Phi = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \tag{4}$$

For discrete-time system, we consider the unit circle, *i.e.*, $\lambda = e^{j\omega}$ and

$$\Phi = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{5}$$

Although the KYP lemma holds for any matrix Φ with negative determinant, our implementation only allows the two cases (4) and (5).

2.2 The generalized KYP lemma

Iwasaki and Hara [IH05] describe several extensions of the KYP lemma that characterize nonnegativity of a rational function on a segment of a straight line or a circle in the complex plane. As above, we assume that $\Phi \in \mathbf{H}^2$ is a matrix with negative determinant and (A, B) is a controllable pair. Suppose $\Psi \in \mathbf{H}^2$ is nonsingular. The generalized KYP lemma states that the FDI (2) holds for all $\lambda \in \mathbf{C}$ that satisfy

$$\begin{bmatrix} \lambda \\ 1 \end{bmatrix}^H \Phi \begin{bmatrix} \lambda \\ 1 \end{bmatrix} = 0, \qquad \begin{bmatrix} \lambda \\ 1 \end{bmatrix}^H \Psi \begin{bmatrix} \lambda \\ 1 \end{bmatrix} \geq 0$$

if and only if there exist a $P, Q \in \mathbf{H}^n$ such that

$$M + \begin{bmatrix} A & B \\ I & 0 \end{bmatrix}^H (\Phi \otimes P + \Psi \otimes Q) \begin{bmatrix} A & B \\ I & 0 \end{bmatrix} \preceq 0, \qquad Q \succeq 0. \tag{6}$$

For example, choosing

$$\Phi = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad \Psi = \begin{bmatrix} -1 & j(\omega_{\mathrm{L}} + \omega_{\mathrm{H}})/2 \\ -j(\omega_{\mathrm{L}} + \omega_{\mathrm{H}})/2 & -\omega_{\mathrm{L}}\omega_{\mathrm{H}} \end{bmatrix},$$

gives a frequency domain condition on a segment $\{j\omega \mid \omega_{\mathrm{L}} \le \omega \le \omega_{\mathrm{H}}\}$ of the imaginary axis. Taking

$$\Phi = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \qquad \Psi = \begin{bmatrix} 0 & e^{j\alpha} \\ e^{-j\alpha} & -2\cos\beta \end{bmatrix},$$

where $0 < \beta < \pi$, gives a frequency domain condition on a segment of the unit circle $\{e^{j\omega} \mid |\omega - \alpha| \le \beta\}$.

2.3 Sampled form of the (generalized) KYP lemma

The `gKYPSDP` solver is based on the following reformulation of the KYP lemma. We can show that (3) holds for some P if and only if there exists an $X \succeq 0$ such that

$$V(\lambda)^H (M + X) V(\lambda) = 0, \quad \forall \lambda \in \mathcal{C}, \tag{7}$$

where $\mathcal{C}$ is a finite set of at least $2n+1$ distinct sample points s_i that satisfy

$$\begin{bmatrix} s_i \\ 1 \end{bmatrix}^H \Phi \begin{bmatrix} s_i \\ 1 \end{bmatrix} = 0.$$

Furthermore, we can show that (6) holds for some P and $Q \succeq 0$ if and only if there exist matrices $X_1 \succeq 0$ and $X_2 \succeq 0$ such that

$$V(\lambda)^H (M + X_1) V(\lambda) + g(\lambda) U(\lambda)^H X_2 U(\lambda) = 0, \quad \forall \lambda \in \mathcal{C}, \tag{8}$$

where

$$g(\lambda) = \begin{bmatrix} \lambda \\ 1 \end{bmatrix}^H \Psi \begin{bmatrix} \lambda \\ 1 \end{bmatrix}.$$

Therefore, the equivalent SDP of (1) after reformulation is

$$\begin{array}{ll} \text{min.} & w^T x \\ \text{s.t.} & V_i(\lambda_i)^H (M_i(x) + N_i + X_{1i}) V_i(\lambda_i) + g_i(\lambda_i) U_i(\lambda_i)^H X_{2i} U_i(\lambda_i) = 0, \quad \forall \lambda_i \in \mathcal{C}_i, \\ & X_{1i} \succeq 0, \quad X_{2i} \succeq 0, \quad i = 1, 2, \ldots, L. \end{array} \tag{9}$$

3 Software description

`gKYPSDP` requires Matlab and the Control System Toolbox. Unzip `gkypsdp.zip` and add the directory (folder) `gkypsdp` to your Matlab path.

3.1 Limitations

The current version has the following limitations.

- $m_i \leq 2$. The system needs to be single-input or two-input. For $m_i = 2$, only matrices B_i of the form
$$B_i = \begin{bmatrix} b_i & 0 \end{bmatrix},$$
are allowed. This limit is imposed because the algorithm is best suited for the single-input case.

- The pairs (A_i, B_i) must be controllable.

- Φ_i is either $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ for continuous-time or $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ for discrete-time.

- $n_i > 0$.

- The solver does not detect infeasibility or unboundedness.

3.2 `gkypsdp_solver.m`

The main routine is `gkypsdp_solver.m`.

```
[sol, info] = gkypsdp_solver(prob, opt);
```

First input argument (`prob`) The argument `prob` is a Matlab structure containing the problem data.

- `prob.L`: the number of constraints L.

- `prob.w`: the cost vector $w \in \mathbf{R}^p$.

- `prob.A{i}`: the matrix $A_i \in \mathbf{R}^{n_i \times n_i}$ in the ith constraint.

- `prob.B{i}`: the matrix $B_i \in \mathbf{R}^{n_i \times m_i}$ in the ith constraint.

- `prob.Phi{i}`: the matrix $\Phi_i \in \mathbf{H}^2$.

- `prob.Psi{i}`: the matrix $\Psi_i \in \mathbf{H}^2$.

- `prob.N{i}`: the matrix $N_i \in \mathbf{H}^{n_i+m_i}$

- `prob.M{i}`: the matrix $[\mathbf{vec}(M_{i1}), \ldots, \mathbf{vec}(M_{ip})]$, where $M_{ij} \in \mathbf{H}^{n_i+m_i}$. Here, $\mathbf{vec}(T)$ denotes the matrix T stored as a vector in column major order. (As obtained, for example, by the Matlab code `T(:)`.)

Second input argument (`opt`) The argument `opt` is optional and specifies algorithm parameters.

- `opt.sample`: specifies the sampling method used in the reformulation of §2.3. The default value is 1. In this case the samples are computed from a Schur decomposition after a balancing transformation [LV07]. If `opt.sample` is equal to 2, uniform sampling is used after of a balancing transformation. If equal to 3, uniform sampling is used with the original state-space model.
- `opt.IPMsolver`: specifies the interior-point method used to solve the problem.. If `opt.IPMsolver` is 1, a custom IPM solver written in Matlab is used. The method exploits low-rank structure in the sampling reformulation (9). If `opt.IPMsolver` is 2, the SeDuMi solver is applied to the original problem formulation (1). If `opt.IPMsolver` is 3, the SDPT3 solver is applied to the the original problem. Options 2 and 3 require Yalmip [Löf04a] and SeDuMi [Stu99], resp., SDPT3 [TTT03]. Default is 1.
- `opt.feasx`: the initial value of x. Used only when `opt.IPMsolver` is 1. The default value is the zero vector.
- `opt.maxiters`: maximum number of iterations. Default is 50.
- `opt.abstol`: absolute tolerance of duality gap. Default is 10^{-6}.
- `opt.reltol`: relative tolerance of duality gap. Default is 10^{-6}.
- `opt.pfeastol`: residual tolerance in the primal feasibility equations. Default is 10^{-6}.
- `opt.dfeastol`: residual tolerance in the dual feasibility equations. Default is 10^{-6}.

First output argument (`sol`) The argument `sol` is a Matlab structure containing the solutions.

- `sol.x`: primal x at the last iteration.
- `sol.z`: dual variables of reformulated SDP (9) at the last iteration.
- `sol.Pobj`: the primal objective value, $w^T x$.
- `sol.Dobj`: the dual objective value of (9).

Second output argument (`info`) The argument `info` is optional and contains solver statistics.

- `info.ptime`: preprocessing time in CPU seconds.
- `info.stime`: IPM solving time in CPU seconds.
- `info.iters`: number of iterations.

- `info.pfeas`: residual in the primal feasibility equations.
- `info.dfeas`: residual in the dual feasibility equations.
- `info.absgap`: absolute gap between primal and dual objectives.
- `info.relgap`: relative gap between primal and dual objectives.

3.3 A simple example

As an example, we use `gKYPSDP` to compute the $\mathbf{H}_\infty$-norm of the transfer function:

$$H(s) = c(sI - A)^{-1}b + d.$$

The solution can be computed by solving the following optimization

$$\begin{array}{ll} \text{minimize} & \gamma^2 \\ \text{subject to} & H(s)^H H(s) \preceq \gamma^2 I. \end{array}$$

This is equivalent to the KYP-SDP

$$\begin{array}{ll} \text{minimize} & \gamma^2 \\ \text{subject to} & \begin{bmatrix} A^T P + PA & PB \\ B^T P & 0 \end{bmatrix} + \begin{bmatrix} c^T & 0 \\ d^T & 1 \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & -\gamma^2 I \end{bmatrix} \begin{bmatrix} c & d \\ 0 & 1 \end{bmatrix} \preceq 0. \end{array}$$

We choose a SISO system with 3 states. The following is the Matlab code for computing the $\mathbf{H}_\infty$ norm.

```
% construct gKYP-SDP
prob.w = [1];
prob.L = 1;
prob.A{1} = A;
prob.B{1} = b;
prob.Phi{1} = [0,1;1,0];
prob.Psi{1} = [0,0;0,0];
prob.N{1} = [c'*c, c'*d; d'*c, d'*d];
prob.M{1}(:,1) = vec([zeros(3,4); zeros(1,3),-1]);

% solve gKYP-SDP
[sol, info] = gkypsdp_solver(prob);

% H infinity norm
gamma = sqrt(sol.x);
```

The complete Matlab code can be found in `ex_NORM.m` in the `examples` directory.

3.4 Caveats

Here we note some known problems with the `gKYPSDP` solver (similar problems are often encountered in general-purpose interior-point SDP solvers as well). Sometimes the solver encounters numerical difficulties in the later stages of the iteration, before the required accuracy is reached. When the solver detects these situations, the solution at the last iteration is often still acceptable. In some cases, a remedy is to increase the tolerances `ops.abstol`, `ops.reltol`, or `ops.feastol`. Numerical problems can often be resolved by changing the initial starting values. `gkypsdp` computes the initial x heuristically; however, the user can specify a different initial value for x, by providing `opt.feasx`.

4 Examples

4.1 Randomly generated KYP-SDPs

We first consider a family of randomly generated problems with one KYP-LMI constraint:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & \begin{bmatrix} A & B \\ I & 0 \end{bmatrix}^T (\Phi \otimes P) \begin{bmatrix} A & B \\ I & 0 \end{bmatrix} + M(x) \preceq 0, \end{array} \tag{10}$$

where $c \in \mathbf{R}^p$, $A \in \mathbf{R}^{n \times n}$ and $B \in \mathbf{R}^n$. The system orders n range from 20 to 1000; the dimension of the variable x is $p = n/5$. The state-space models are constructed by randomly generating orthogonal matrices

$$\left[\begin{array}{cc} A & B \\ C & D \end{array} \right].$$

We choose Φ to be (5). The coefficients of the linear mapping $M(x)$ are randomly generated in such a way that the problem is strictly feasible. Instances that are dual infeasible (unbounded below) are discarded.

The table shows the CPU times in seconds on a 3.0 GHz Pentium 4 with 3.0 GB of memory, using Matlab 7.4 (R2007a). All times are averaged over five randomly generated instances. The number of iterations itself is not reported but was roughly 10–15 for all the algorithms. Blank entries in the table indicate that the simulation was aborted due to excessive execution time or an out-of-memory error.

Columns 2 and 3 show the times per iteration for solving the inequality form SDP (10) using the general-purpose solver SeDuMi (version 1.1R3) and SDPT3 (version 4.0 beta) [TTT03], via the YALMIP interface [Löf04b]. The YALMIP pre-processing time was excluded when calculating the times per iteration. Column 4 and 5 show the times per iteration using the SeDuMi and SDPT3 solvers directly for solving the equivalent standard form SDP

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & V_*(z)(M(x) + X)V(z) = 0 \;\; \forall z \in \mathcal{C} \\ & X \succeq 0 \end{array} \tag{11}$$

	Inequality form		Standard form		KYPD	gKYPSDP
n	SeDuMi	SDPT3	SeDuMi	SDPT3		
20	0.12	0.20	0.03	0.03	0.02	0.01
30	0.72	0.59	0.05	0.05	0.04	0.02
45	5.7	3.3	0.14	0.10	0.09	0.04
70	64	33	0.63	0.31	0.41	0.12
100			2.6	0.91	1.2	0.31
150			10	3.6	4.7	0.94
220			44	15	18	2.9
350						11.1
500						30.2
750						98.2
1000						227

Table 1: Times per iteration (sec.) of different solvers for KYP-SDPs.

where $\mathcal{C}$ is a set of $2n+1$ sample points on the unit circle, generated as described in [LV07]. The next column, labeled KYPD, shows the results for the KYPD Matlab package [Wal03], which implements the algorithm of [VBW+05]. This method requires a significant amount of processing before the start of the first iteration, and we excluded the preprocessing time when calculating the time per iteration. The last column shows the results of `gKYPSDP`.

Figure 1 shows the average times per iteration versus n. We can note that the complexity of the fast algorithm is almost exactly $O(n^3)$.

4.2 Randomly generated gKYP-SDPs

The second experiment (table 2 and figure 2) is based on a family of SDPs with one generalized KYP constraint,

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & \begin{bmatrix} A & B \\ I & 0 \end{bmatrix}^H (\Phi \otimes P + \Psi \otimes Q) \begin{bmatrix} A & B \\ I & 0 \end{bmatrix} + M(x) \preceq 0 \\ & Q \succeq 0. \end{array}$$

The problem is strictly feasible by construction. Dual infeasible problems are discarded. The system orders range from $n = 20$ to $n = 750$, and $p = n/5$.

4.3 Linear-phase band-pass FIR filter design

We consider the problem of designing a linear-phase band-pass FIR filter with frequency response

$$\begin{aligned} H(\omega) &= x_n e^{jn\omega} + \cdots + x_1 e^{j\omega} + x_0 + x_1 e^{-j\omega} + \cdots + x_n e^{-jn\omega} \\ &= x_0 + 2\Re(x_1 e^{-j\omega} + \cdots + x_n e^{-jn\omega}), \end{aligned}$$

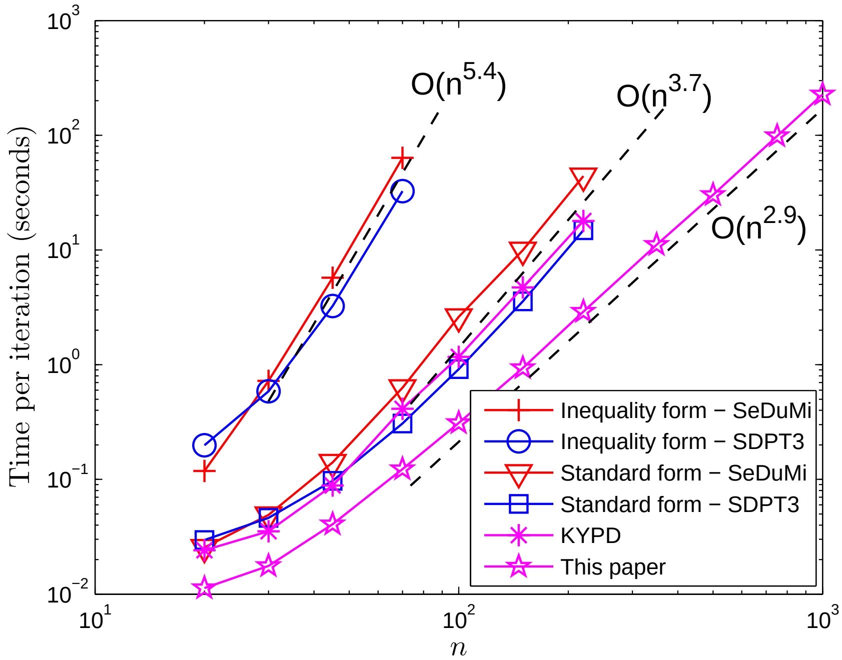


Figure 1: Graph of the results in table 1.

n	Inequality form SeDuMi	Inequality form SDPT3	Standard form SeDuMi	Standard form SDPT3	gKYPSDP
20	0.89	1.2	0.08	0.16	0.02
30	8.0	6.1	0.18	0.17	0.03
45	77	44	0.71	0.54	0.09
70			3.8	2.1	0.24
100			11	7.5	0.61
150			60	38	1.9
220					5.7
350					23
500					63
750					205

Table 2: Times per iteration (sec.) of solvers for gKYP-SDPs.

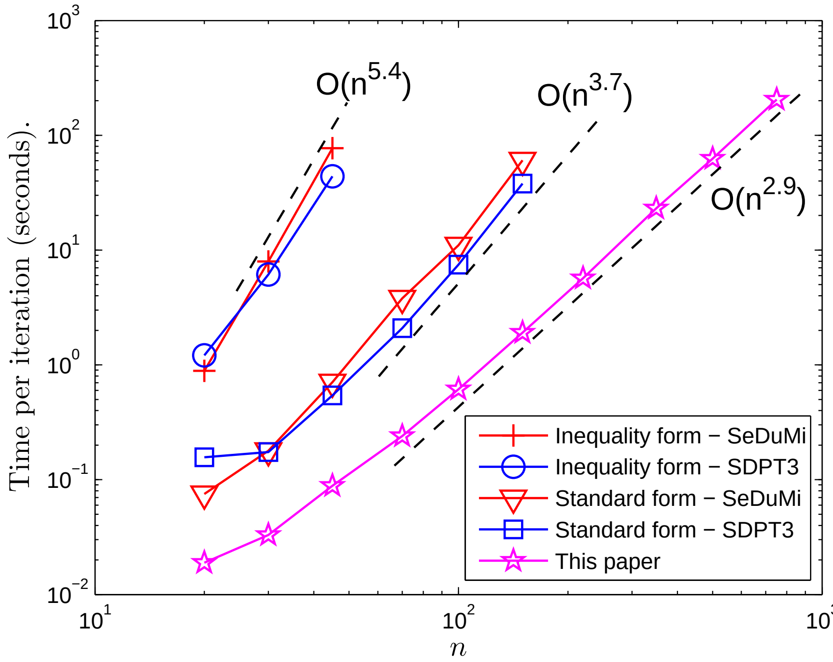


Figure 2: Graph of the result in table 2.

which meets the following design specifications:

1. $|H(\omega)| \le t_{\mathrm{s}}$ for $0 \le \omega \le \omega_{\mathrm{s1}}$ and $\omega_{\mathrm{s2}} \le \omega \le \pi$,
2. $|H(\omega) - 1| \le t_{\mathrm{p}}$ for $\omega_{\mathrm{p1}} \le \omega \le \omega_{\mathrm{p2}}$.

The first requirement constrains the filter gain to be small at both low and high frequency range (stop band). The second requirement ensures the filter has gain close to one in the middle frequency range (pass band). A similar example can be found in [RV06]. We use the state-space model

$$A = \begin{bmatrix} 0 & 0 \\ I_{n-1} & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix}, \quad D = \frac{1}{2}x_0,$$

so

$$G(\omega) = C(e^{j\omega}I - A)^{-1}B + D = \frac{1}{2}x_0 + x_1 e^{-j\omega} + \ldots + x_n e^{-jn\omega}.$$

Let

$$M(x) + N = \begin{bmatrix} C & D \\ 0 & 1 \end{bmatrix}^T \begin{bmatrix} 0 & \alpha \\ \alpha & \beta \end{bmatrix} \begin{bmatrix} C & D \\ 0 & 1 \end{bmatrix},$$

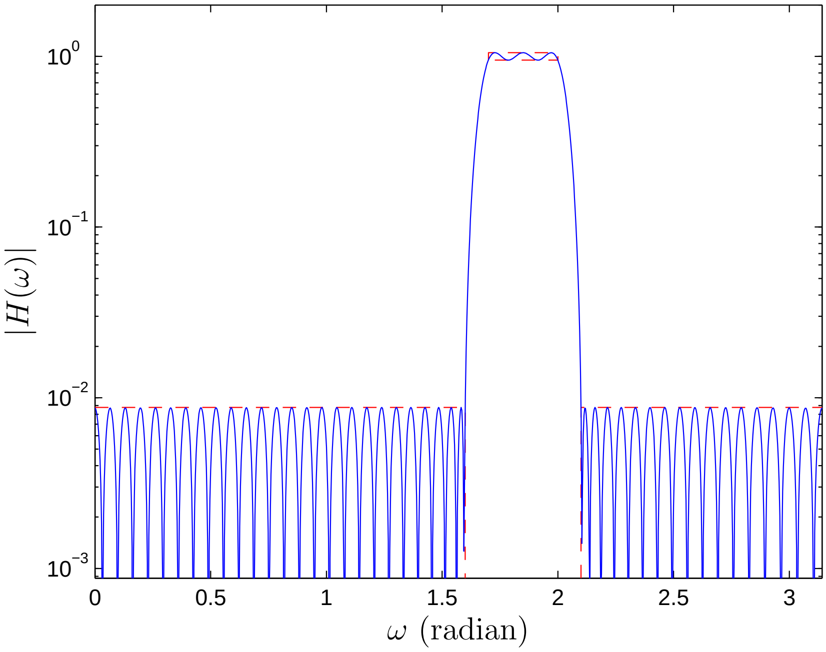


Figure 3: Gain of the band-pass filter with $n = 50$ taps.

then the frequency domain inequality is $\alpha H(\omega) + \beta \leq 0$, where ω is in a finite frequency range. We consider the design problem

$$\begin{array}{ll} \text{minimize} & t_{\text{s}} \\ \text{subject to} & H(\omega) \geq -t_{\text{s}}, \quad 0 \leq \omega \leq \pi \\ & H(\omega) \leq t_{\text{s}}, \quad 0 \leq \omega \leq \omega_{\text{s1}} \\ & H(\omega) \leq t_{\text{s}}, \quad \omega_{\text{s2}} \leq \omega \leq \pi \\ & H(\omega) \geq 1 - t_{\text{p}}, \quad \omega_{\text{p1}} \leq \omega \leq \omega_{\text{p2}} \\ & H(\omega) \leq 1 + t_{\text{p}}, \quad \omega_{\text{p1}} \leq \omega \leq \omega_{\text{p2}}. \end{array}$$

Figure 3 shows an example with

$$n = 50, \qquad \omega_{\text{s1}} = 1.6, \qquad \omega_{\text{s2}} = 2.1, \qquad \omega_{\text{p1}} = 1.7, \qquad \omega_{\text{p2}} = 2.0, \qquad t_{\text{p}} = 0.05.$$

The optimal $t_{\text{s}} = 0.0088$. The Matlab file for this example is `ex_BPF.m` in the `examples` directory.

4.4 Low-pass FIR filter design

This example is taken from [IH05]. We design a nonlinear-phase low-pass FIR filter with frequency response

$$H(\omega) = x_0 + x_1 e^{-j\omega} + \cdots + x_n e^{-jn\omega},$$

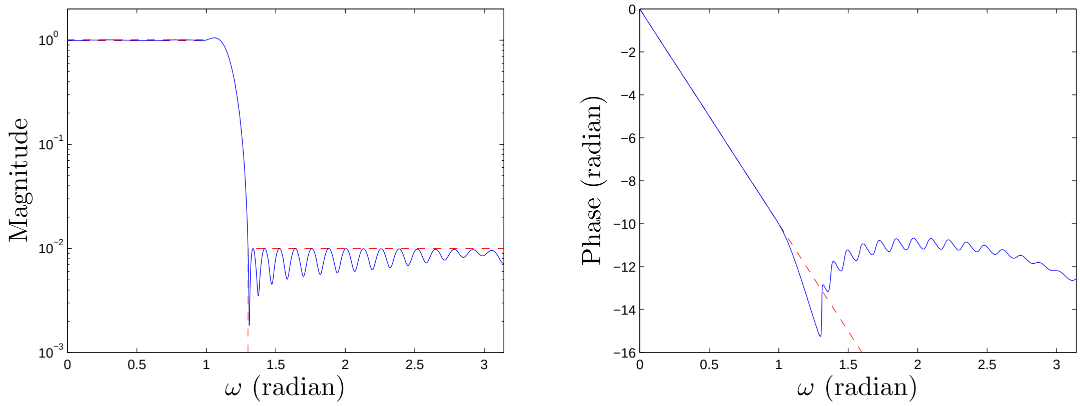


Figure 4: Magnitude and phase of the low-pass filter with $n = 50$ taps and $d = 10$ group delay.

which meets the following design specifications:

1. $|H(\omega)| \le t_{\rm s}$ for $\omega_{\rm s} \le \omega \le \pi$,
2. $|H(\omega) - e^{-jd\omega}| \le t_{\rm p}$ for $0 \le \omega \le \omega_{\rm p}$.

The first requirement constrains the filter gain to be small in the stop band. The second specification requires the filter frequency response to be closed to a desired function $e^{-jd\omega}$ in the pass band. $e^{-jd\omega}$ has unity gain and linear phase with group delay d, where d is an integer in the range $0 < d \le n$. We use the state-space model

$$A = \begin{bmatrix} 0 & 0 \\ I_{n-1} & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix}, \quad D = x_0,$$

so

$$H(\omega) = C(e^{j\omega}I - A)^{-1}B + D.$$

We consider the design problem

$$\begin{array}{ll} \text{minimize} & t_{\rm p} \\ \text{subject to} & |H(\omega)|^2 \le t_{\rm s}^2, \quad \omega_{\rm s} \le \omega \le \pi \\ & |H(\omega) - e^{-jd\omega}|^2 \le t_{\rm p}^2, \quad 0 \le \omega \le \omega_{\rm p}. \end{array}$$

Figure 4 shows an example with

$$n = 50, \qquad w_{\rm p} = 1.0, \qquad \omega_{\rm s} = 1.3, \qquad t_{\rm s} = 0.01, \qquad d = 10.$$

The optimal $t_{\rm p} = 0.0099$. The Matlab file for this example is `ex_LPF.m` in the `examples` directory.

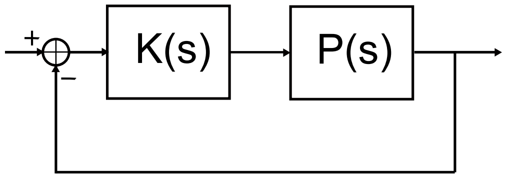


Figure 5: Standard negative-unit feedback system.

4.5 PID controller design

This example is based on the material presented in [HIS06]. The goal is to design a PID controller, as shown in Figure 5,

$$K(s) = k_{\mathrm{p}} + \frac{k_{\mathrm{i}}}{s} + \frac{k_{\mathrm{d}} s}{1 + T_{\mathrm{s}} s},$$

for a plant $P(s)$ such that the open-loop transfer function $L(s)$ meets the following design specifications:

1. $|L(\omega)| \leq t_{\mathrm{h}}$ for $\omega_{\mathrm{h}} \leq \omega \leq \infty$,
2. $a\Re(L(\omega)) + b\Im(L(\omega)) + c \leq 0$ for $0 \leq \omega \leq \infty$,
3. $\Im(L(\omega)) \leq -t_{\mathrm{l}}$ for $0 \leq \omega \leq \omega_{\mathrm{l}}$.

The first requirement with small $t_{\mathrm{h}} > 0$ ensures robustness against un-modeled dynamics, which typically exists in the high frequency range. The second requirement guarantees a certain stability margin. The third specification with large $t_{\mathrm{l}} > 0$ requires a high gain at low frequency to reduce steady-state offset and improve disturbance rejection. This approach of constraining the open-loop transfer function is similar to the concept of loop shaping discussed in [KG96, Page 134].

We express $K(s)$ in the controller canonical form

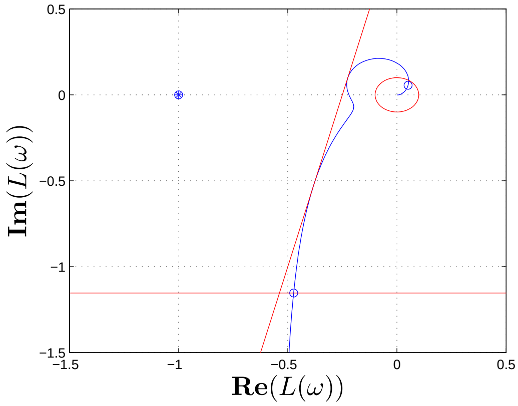
$$K(s) = \left[\begin{array}{c|c} A_k & B_k \\ \hline C_k & D_k \end{array}\right] = \left[\begin{array}{cc|c} 0 & 1 & 0 \\ 0 & -\frac{1}{T_{\mathrm{s}}} & 1 \\ \hline \frac{k_{\mathrm{i}}}{T_{\mathrm{s}}} & k_{\mathrm{i}} - \frac{k_{\mathrm{d}}}{T_{\mathrm{s}}^2} & k_{\mathrm{p}} + \frac{k_{\mathrm{d}}}{T_{\mathrm{s}}} \end{array}\right],$$

where T_s is a time constant for approximating the derivative term by a rational function. The plant we consider has the transfer function

$$P(s) = \frac{10}{(s+1)(s^2+s+10)(s^2+4s+15)}.$$

The open-loop transfer function, $L(s) = P(s)K(s)$, has the state-space realization

$$L(s) = \left[\begin{array}{cc|c} A_p & 0 & B_p \\ B_k C_p & A_k & B_k D_p \\ \hline D_k C_p & C_k & D_k D_p \end{array}\right].$$



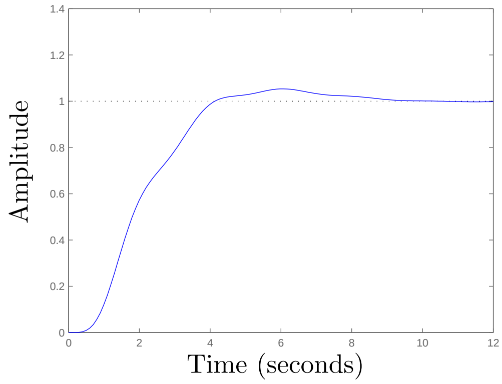


Figure 6: PID controller design: the left graph is the Nyquist plot of the open-loop transfer function $L(\omega)$; the right graph is the step response.

We consider the design problem

$$\begin{array}{ll} \text{minimize} & -t_{\mathrm{l}} \\ \text{subject to} & |L(\omega)| \leq t_{\mathrm{h}}, \quad \omega_{\mathrm{h}} \leq \omega \leq \infty \\ & a\Re(L(\omega)) + b\Im(L(\omega)) + c \leq 0, \quad 0 \leq \omega \leq \infty \\ & \Im(L(\omega)) \leq -t_{\mathrm{l}}, \quad 0 \leq \omega \leq \omega_{\mathrm{l}}. \end{array}$$

Figure 6 shows an example with

$$\omega_{\mathrm{l}} = 0.4, \qquad \omega_{\mathrm{h}} = 4, \qquad t_{\mathrm{h}} = 0.1, \qquad a = -4, \qquad b = 1, \qquad c = -1, \qquad T_{\mathrm{s}} = 0.1.$$

The optimal parameters are

$$k_{\mathrm{p}} = 2.7282, \qquad k_{\mathrm{i}} = 7.9185, \qquad k_{\mathrm{d}} = 0.6371, \qquad t_{\mathrm{l}} = 1.1531.$$

The Matlab file for this example is `ex_PID.m` in the `examples` directory.

4.6 Robust stabilizing controller synthesis

This example is taken from [HSK03]. Consider the standard negative-unit feedback system, shown in Figure 5, with the plant

$$P(\lambda) = \frac{b(\lambda)}{a(\lambda)} = \frac{b_0 + b_1\lambda + \cdots + b_n\lambda^n}{a_0 + a_1\lambda + \cdots + a_n\lambda^n}$$

and controller $K(s)$

$$K(\lambda) = \frac{y(\lambda)}{x(\lambda)} = \frac{y_0 + y_1\lambda + \cdots + y_m\lambda^m}{x_0 + x_1\lambda + \cdots + x_m\lambda^m}.$$

Assume that the plant has structured parametric uncertainty: the transfer function $b(\lambda)/a(\lambda)$ belongs to a polytope with N given vertices $b^1(\lambda)/a^1(\lambda), \ldots, b^N(\lambda)/a^N(\lambda)$. Then the characteristic polynomial

$$c(\lambda) = a(\lambda)x(\lambda) + b(\lambda)y(\lambda)$$

is of degree $m+n$ with vertices $c^i(\lambda) = a^i(\lambda)x(\lambda) + b^i(\lambda)y(\lambda)$ for $i = 1, \ldots, N$. We define the stable region

$$R = \{\lambda \in \mathbf{C} : \begin{bmatrix} \lambda^H & 1 \end{bmatrix} \Phi \begin{bmatrix} \lambda \\ 1 \end{bmatrix} < 0\}.$$

Let $d(\lambda)$ be a R-stable polynomial of degree $n+m$. The sufficient condition [HSK03] for robust R-stable controller synthesis is the existence of $P_i \in \mathbf{H}^{n+m}$ such that

$$\begin{bmatrix} A & B \\ I & 0 \end{bmatrix}^H (\Phi \otimes P_i) \begin{bmatrix} A & B \\ I & 0 \end{bmatrix} + \begin{bmatrix} c^i \\ d \end{bmatrix}^H \begin{bmatrix} 0 & 1 \\ 1 & -2\gamma \end{bmatrix} \begin{bmatrix} c^i \\ d \end{bmatrix} \succeq 0, \quad i = 1, \ldots, N,$$

where

$$A = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

and c^i, d are row vectors containing the coefficients of the polynomial with ascending degree orders. γ is an arbitrarily small positive number.

To illustrate the technique we use an ARMAZ model of a PUMA 762 robotic disk grinding process [TS94]. Due to the nonlinearity of the system plant, parameters can vary up to 20% from its nominal values. The discrete-time plant transfer function is

$$\frac{b(z^{-1})}{a(z^{-1})} = \frac{(-0.1688 + q_4)z^{-3} + (-0.1619 + q_3)z^{-2} + (-0.0764 + q_2)z^{-1} + (0.0257 + q_1)}{0.2508z^{-4} - 1.0265z^{-3} + 1.779z^{-2} - 1.914z^{-1} + 1},$$

where $|q_4| \leq 0.03376$, $|q_3| \leq 0.03238$, $|q_2| \leq 0.01528$, and $|q_1| \leq 0.00514$. Since there are 4 varying parameters, so the number of vertices is $N = 2^4 = 16$. The characteristic polynomial is

$$c(z) = z^{12}\left((1 - z^{-1})a(z^{-1})x(z^{-1}) + z^{-5}b(z^{-1})y(z^{-1})\right),$$

where $1 - z^{-1}$ in the controller denominator is added to maintain zero steady-state error. The stable central polynomial and stability matrix are chosen to be

$$d(z) = z^{19} \quad \text{and} \quad \Phi = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

A feasible robust controller is obtained

$$\frac{y(z^{-1})}{x(z^{-1})} = \frac{-2.27 + 3.42z^{-1} - 2.81z^{-2} + 1.64z^{-3} - 0.18z^{-4} + 0.16z^{-5} - 0.16z^{-6} - 0.024z^{-7}}{1 + 2.62z^{-1} + 3.8z^{-2} + 3.9z^{-3} + 3.15z^{-4} + 2.18z^{-5} + 1.23z^{-6} + 0.46z^{-7}}.$$

Figure 7 shows the robust root locus. The black dots are the root locii computed from 500 random plants within the uncertainty polytope. The red dots are the root locii computed from the 16 plant vertices. The blue dots are the root locus using the nominal plant. The Matlab file for this example is `ex_RSC.m` in the `examples` directory.

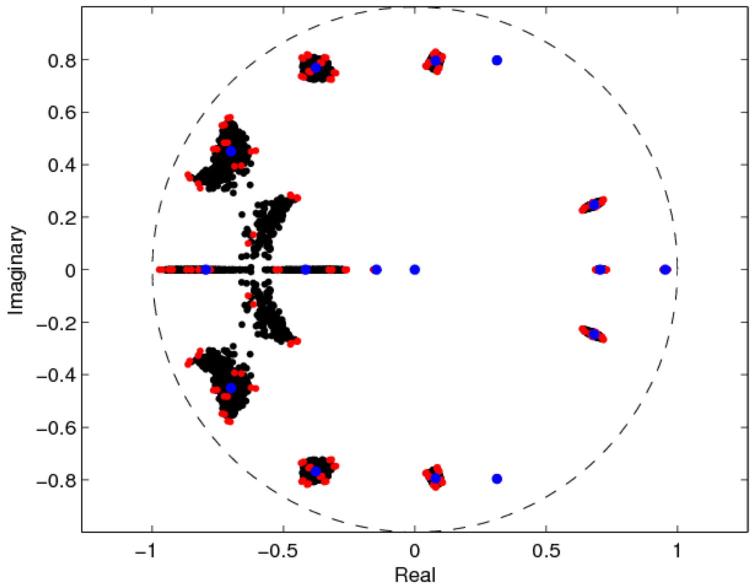


Figure 7: Robust stabilizing controller design: root locus of PUMA 762 robotic disk grinding process

Acknowledgements

We thank Didier Henrion for providing the PUMA 762 robotic disk grinding example. Thanks also go to Goele Pipeleers for feedback on an earlier version of `gKYPSDP` used in optimal controller design for periodic inputs.

Feedback

Any comments, suggestions and reports of applications of `gKYPsdp` are greatly appreciated. Please send feedback to Zhang Liu (`zhang@ee.ucla.edu`) or Lieven Vandenberghe (`vandenbe@ee.ucla.edu`).

References

[HIS06] S. Hara, T. Iwasaki, and D. Shiokata. Robust PID control using generalized KYP synthesis: direct open-loop shaping in multiple frequency ranges. *Control System Magazine*, 26(1):80–91, Feb. 2006.

[HSK03] D. Henrion, M. Sebek, and V. Kucera. Positive polynomials and robust stabilization with fixed-order controllers. *IEEE Transactions on Automatic Control*, 48(7):1178–1186, Jul 2003.

[IH05] T. Iwasaki and S. Hara. Generalized KYP lemma: unified frequency domain inequalities with design applications. *IEEE Transactions on Automatic Control*, 50(1):41–59, 2005.

[KG96] K. Zhou with J. Doyle and K. Glover. *Robust and Optimal Control*. Prentice Hall, 1996.

[KM07] C.-Y. Kao and A. Megretski. On the new barrier function and specialized algorithms for a class of semidefinite programs. *SIAM Journal on Control and Optimization*, 46(2):468–495, 2007.

[Löf04a] J. Löfberg. *YALMIP : A Toolbox for Modeling and Optimization in MATLAB*, 2004.

[Löf04b] J. Löfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[LV07] Z. Liu and L. Vandenberghe. Low-rank structure in semidefinite programs derived from the KYP lemma. In *Proceedings of the 46th IEEE Conference on Decision and Control*, 2007. To appear.

[RV06] T. Roh and L. Vandenberghe. Discrete transforms, semidefinite programming and sum-of-squares representations of nonnegative polynomials. *SIAM Journal on Optimization*, 16:939–964, 2006.

[Stu99] J. F. Sturm. Using SEDUMI 1.02, a Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11-12:625–653, 1999.

[TS94] Y. Tong and N.K. Sinha. A computational technique for the robust root locus. *IEEE Transactions on Industrial Electronics*, 41(1):79–85, Feb 1994.

[TTT03] R. H. Tütüncü, K. C. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical Programming Series B*, 95:189–217, 2003.

[VBW+05] L. Vandenberghe, V. R. Balakrishnan, R. Wallin, A. H. Hanson, and T. Roh. Interior point algorithms for semidefinite programming problems derived from the KYP lemma. In D. Henrion and A. Garulli, editors, *Positive Polynomials in Control*, volume 312 of *Lecture Notes on Control and Information Sciences*. Springer Verlag, 2005.

[Wal03] R. Wallin. User's guide to kypd_solver. Technical Report LiTH-ISY-R-2517, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, 2003.