User’s guide to kypd_solver

Ragnar Wallin

Automatic control
Department of Electrical Engineering
Linköpings universitet, SE-581 83 Linköping, Sweden
WWW: http://www.control.isy.liu.se
E-mail: ragnarw@isy.liu.se

April 23, 2003

Report no.: LiTH-ISY-R-2517

Technical reports from the Control & Communication group in Linköping are available at http://www.control.isy.liu.se/publications.
User’s guide to \texttt{kypd\_solver}

Ragnar Wallin

April 23, 2003

1 Introduction

This package contains software for solving semidefinite programs (SDPs) originating from the Kalman-Yakubovich-Popov (KYP) lemma. These SDPs have the following structure

\[
\begin{align*}
\text{minimize} & \quad c^T x \\
\text{subject to} & \quad X_i = \begin{bmatrix} P_i A_i + A_i^T P_i & P_i B_i \\
B_i^T P_i & 0 \end{bmatrix} + M_{i0} + \sum_{j=1}^{K} x_j M_{ij} \succeq 0 \quad (1)
\end{align*}
\]

The problem data are the vector \(c \in \mathbb{R}^K\), the matrices \(A_i \in \mathbb{R}^{n_i \times n_i}\), \(B_i \in \mathbb{R}^{n_i \times m_i}\) and the symmetric matrices \(M_{ij}\). The inequality \(X_i \succeq 0\) means that \(X_i\) is positive semidefinite. To make the code efficient the dual problem is solved using SeDuMi [2] after a reduction of the number of variables. The SDP is formulated using YALMIP [3]. The primal variables \(P\) and \(x\) are reconstructed afterwards. For more information about this see [1].

2 Matlab routines

The main routines are \texttt{kypd.m} and \texttt{kypd\_solver.m}. To solve a given SDP you can proceed as follows:

1. if you know that the \(M_{ij}\)-matrices are linearly independent, the pairs \((A_i, B_i)\) are controllable and all \(A_i\)s are Hurwitz use \texttt{kypd.m}. This will save some time used for checking the above mentioned properties.

2. otherwise use \texttt{kypd\_solver.m}.
kypd.m

[u,P,x]=kypd(matrix_info,tol,maxiters)

Purpose

The function kypd solves the problem

\[
\begin{align*}
\text{minimize} & \quad c^T x \\
\text{subject to} & \quad X_i = \begin{bmatrix} P_i A_i + A_i^T P_i & P_i B_i \\ B_i^T P_i & 0 \end{bmatrix} + M_{i0} + \sum_{j=1}^{K} x_j M_{ij} \geq 0 \\
& \quad i = 1, 2, \ldots, N
\end{align*}
\]

Input arguments

All matrix input information is stored in a cell structure called matrix_info. All fields are described below along with the other input arguments.

1. • matrix_info.N: Number of constraints.
   • matrix_info.K: Number of elements in the vector x of the primal objective function.
   • matrix_info.c: The vector c in the primal objective.
   • matrix_info.A{i}: The A-matrix appearing in constraint i of the primal problem.
   • matrix_info.B{i}: The B-matrix appearing in constraint i of the primal problem.
   • matrix_info.M0{i}: The M_0-matrix appearing in constraint i of the primal problem.
   • matrix_info.M{i,j}: The jth M-matrix appearing in constraint i of the primal problem.
   • matrix_info.n: A vector containing the sizes of the A-matrices, i.e. \( n = [n_1, n_2, \ldots, n_N]^T \).
   • matrix_info.nm: A vector containing the sizes of the M-matrices, i.e. \( nm = [nm_1, nm_2, \ldots, nm_N]^T \). If \( A_i \in \mathbb{R}^{n_i \times m_i} \) and \( B_i \in \mathbb{R}^{n_i \times m_i} \) then \( nm_i = n_i + m_i \).
2. \( \texttt{tol} \): The solution will not violate feasibility and optimality conditions with more than \( \texttt{tol} \). See \texttt{par.eps} in the SeDuMi user’s manual [2] for details. Default value \( 10^{-8} \).

3. \( \texttt{maxiters} \): Maximum number of iterations. \( \texttt{maxiters} \geq 0 \). Default value 100.

**Output arguments**

1. \( \texttt{u} \): The primal objective \( c^T x \) is stored in \( u \).
2. \( \texttt{P} \): The primal variable \( P \).
3. \( \texttt{x} \): The primal variable \( x \).

**Caveats**

- The matrices \( M_{ij} \) must be linearly independent and the pairs \( (A_i, B_i) \) must be controllable. If this is not the case the problem is not dual strictly feasible or can be reduced to a problem with fewer variables. Additionally, all \( A_i \)-matrices must be Hurwitz.

\texttt{kypd\_solver.m}

\[ \texttt{[u,P,x]=IQC\_solver(matrix\_info,transform,rho,tol,maxiters)} \]

**Purpose**

To solve the KYP based SDP using a primal-dual method the \( M_{ij} \)-matrices have to be linearly independent, the pairs \( (A_i, B_i) \) have to be controllable and the \( A_i \)-s Hurwitz. If the matrices \( M_{ij} \) are linearly dependent there are two possibilities. Either the problem is not dual strictly feasible and can be reduced to a problem with fewer variables which is dual strictly feasible or the primal is unbounded from below. If the system is stabilizable but not controllable there does not exist a strictly feasible dual point. Also in this case the problem can be reduced to a problem with fewer variables which is controllable and for which a strictly feasible dual point exists. If the system is not Hurwitz state feedback can be applied to make it so. Sometimes we
want to apply feedback anyway to possibly improve numerical properties. `kypd_solver` will handle all those problems and produce a correct solution to the original problem, that is the problem before reduction and feedback.

**Feedback**

If the system is not Hurwitz or if the input argument `transform=1` a congruence transformation is done on (1).

\[
\tilde{X}_i = \begin{bmatrix} I & 0 \\ -L_i & I \end{bmatrix}^T X_i \begin{bmatrix} I & 0 \\ -L_i & I \end{bmatrix}
\]

where \( L_i = \frac{1}{\rho} B_i^T S_i \) and \( S_i \) is the solution to the algebraic Riccati equation

\[
S_i A_i + A_i^T S_i - \rho S_i B_i B_i^T S_i + I = 0
\]

The resulting new \( \tilde{A}_i = (A_i - B_i L_i) \) is Hurwitz.

**Input arguments**

1. `matrix_info`: Information about the matrices in the problem. See the input arguments for `kypd` for details.

2. `transform`: If `transform=0` state feedback is only done when the system is not Hurwitz. If `transform=1` state feedback is done even if it is not necessary. It may improve numerical properties. Default value is 0.

3. `rho`: A nonnegative parameter that is used for the state feedback. Default value is 1.

4. `tol`: The solution will not violate feasibility and optimality conditions with more than `tol`. See `par.eps` in the SeDuMi user’s manual [2] for details. Default value is \( 10^{-8} \).

5. `maxiters`: The maximum number of iterations. Default value is 100.

If a default value of an input parameter is wanted it can be put to the empty matrix `[]`.
Output arguments

1. u: The primal objective value $c^T x$ is stored in $u$.

2. P: The primal variable $P$.

3. x: The primal variable $x$.

3 Installation

Install SeDuMi [2] and YALMIP [3] according to the manuals. Unzip the file kypd.zip in the directory where you want to place the matlab files for solving the KYP-based SDPs.

References


http://fewcal.kub.nl/sturm/software/sedumi.html

http://www.control.isy.liu.se/ johanl/yalmip.html