

Efficient implementation of interior-point methods for SDPs arising in control and signal processing

Lieven Vandenberghe

UCLA Electrical Engineering Department

joint work with:

Ragu Balakrishnan (Purdue University)

Anders Hansson, Ragnar Wallin (Linköping University)

IMA Workshop on SDP and Robust Optimization 3/13/03

Outline

- SDPs derived from the Kalman-Yakubovich-Popov (KYP) lemma
- primal-dual interior-point methods
- fast implementation for KYP-SDPs and numerical examples

SDPs derived from the KYP lemma

minimize $c^T x + \sum_{k=1}^N \mathbf{Tr}(C_k P_k)$

subject to $\begin{bmatrix} A_k^T P_k + P_k A_k & P_k B_k \\ B_k^T P_k & 0 \end{bmatrix} + \sum_{i=1}^p x_i M_{ki} \succeq N_k, \quad k = 1, \dots, N$

- variables: $x \in \mathbf{R}^p$, $P_k \in \mathbf{S}^{n_k}$, $k = 1, \dots, N$
- $c \in \mathbf{R}^p$, $C_k \in \mathbf{S}^{n_k}$, $A_k \in \mathbf{R}^{n_k \times n_k}$, $B_k \in \mathbf{R}^{n_k \times m_k}$, $M_{ki}, N_k \in \mathbf{S}^{n_k + m_k}$
- N LMIs of dimension $n_k + m_k$

Kalman-Yakubovich-Popov lemma

if (A, B) is controllable, then the following conditions are equivalent:

1. the KYP-LMI

$$\begin{bmatrix} AP + PA & PB \\ B^T P & 0 \end{bmatrix} + \sum_{i=1}^p x_i M_i - N \succeq 0$$

is feasible (an LMI with variables P, x)

2. the frequency-domain inequality

$$\begin{bmatrix} (j\omega I - A)^{-1} B \\ I \end{bmatrix}^* \left(\sum_{i=1}^p x_i M_i - N \right) \begin{bmatrix} (j\omega I - A)^{-1} B \\ I \end{bmatrix} \succeq 0 \quad \forall \omega \in \mathbf{R}$$

is feasible, where $j = \sqrt{-1}$ (an infinite # of LMIs in x)

Practical importance of KYP-SDPs

minimize $c^T x + \sum_{k=1}^N \mathbf{Tr}(C_k P_k)$

subject to $\begin{bmatrix} A_k^T P_k + P_k A_k & P_k B_k \\ B_k^T P_k & 0 \end{bmatrix} + \sum_{i=1}^p x_i M_{ki} \succeq N_k, \quad k = 1, \dots, N$

- a useful class of SDPs, widely encountered in control
- x is usually the optimization variable of interest; variables P_k are auxiliary variables, introduced to convert semi-infinite frequency-domain inequalities into LMIs
- large number of variables (several 1000 or 10,000) for moderate values of n_k

Algorithms for KYP-SDPs

- general-purpose SDP solvers: slow due to large number of variables
- cutting-plane algorithms (Kao and Megretski, Parrilo)
- fast implementation of standard IPMs using conjugate gradients (Hansson)
- this talk: fast implementation of standard IPMs using direct linear algebra

Special case: Riccati inequality

maximize $\text{Tr } P$

$$\text{subject to } \begin{bmatrix} A^T P + PA & PB \\ B^T P & 0 \end{bmatrix} \preceq \begin{bmatrix} -Q & 0 \\ 0 & -I \end{bmatrix}$$

$(Q \succ 0)$

- a KYP-SDP with $p = 0$ (no variable x), $N = 1$ (one constraint)
- cost of solving: $O(n^3)$, via Schur decomposition of Hamiltonian matrix

$$H = \begin{bmatrix} A & -BB^T \\ -Q & -A^T \end{bmatrix}$$

Properties of Hamiltonian matrix

$$H = \begin{bmatrix} A & -BB^T \\ -Q & -A^T \end{bmatrix}$$

- eigenvalues are symmetric about the imaginary axis; no eigenvalues on the imaginary axis
- if (V_1, V_2) spans the stable invariant subspace, *i.e.*,

$$\begin{bmatrix} A & -BB^T \\ -Q & -A^T \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \bar{A}$$

with \bar{A} stable, then:

- V_1 is nonsingular; $V_2^T V_1 = V_1^T V_2$
- $P_{\text{opt}} = V_2 V_1^{-1}$ is the optimal solution of the SDP

proof: use congruence $T = \begin{bmatrix} V_1 & 0 \\ -B^T V_2 & I \end{bmatrix}$ to transform SDP

maximize $\mathbf{Tr} P$

subject to $T^T \begin{bmatrix} A^T P + PA & PB \\ B^T P & 0 \end{bmatrix} T \succeq T^T \begin{bmatrix} -Q & 0 \\ 0 & -I \end{bmatrix} T$

simplifications (using definition of V_1, V_2):

$$T^T \begin{bmatrix} A^T P + PA & PB \\ B^T P & 0 \end{bmatrix} T = \begin{bmatrix} \bar{A}^T \bar{P} + \bar{P} \bar{A} & \bar{P} \bar{B} \\ \bar{B}^T \bar{P} & 0 \end{bmatrix}$$

$$T^T \begin{bmatrix} -Q & 0 \\ 0 & -I \end{bmatrix} T = \begin{bmatrix} \bar{A}^T P_0 + P_0 \bar{A} & P_0 \bar{B} \\ \bar{B}^T P_0 & -I \end{bmatrix}$$

where $\bar{P} = V_1^T P V_1$, $P_0 = V_1^T V_2$, $\bar{B} = V_1^{-1} B$

optimality conditions of SDP after congruence transformation:

1. primal feasibility: $S \succeq 0$,

$$- \begin{bmatrix} S_{11} & S_{12} \\ S_{12}^T & S_{22} \end{bmatrix} + \begin{bmatrix} \bar{A}^T \bar{P} + \bar{P} \bar{A} & \bar{P} \bar{B} \\ \bar{B}^T \bar{P} & 0 \end{bmatrix} = \begin{bmatrix} \bar{A}^T P_0 + P_0 \bar{A} & P_0 \bar{B} \\ \bar{B}^T P_0 & -I \end{bmatrix}$$

2. dual feasibility: $Z \succeq 0$, $\bar{A}Z_{11} + Z_{11}\bar{A}^T + \bar{B}Z_{12}^T + Z_{12}\bar{B}^T + V_1V_1^T = 0$

3. complementary slackness: $\mathbf{Tr}(SZ) = 0$

solution

$$\bar{P} = P_0, \quad S = \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix}, \quad Z = \begin{bmatrix} Z_{11} & 0 \\ 0 & 0 \end{bmatrix}$$

where $\bar{A}Z_{11} + Z_{11}\bar{A}^T + V_1V_1^T = 0$

SDP duality

primal SDP

$$\begin{array}{ll} \text{minimize} & \langle c, y \rangle \\ \text{subject to} & \mathcal{A}(y) + S = B, \quad S \succeq 0 \end{array}$$

- $\mathcal{A} : \mathcal{V} \rightarrow \mathbf{S}^m$ is a linear mapping; $\langle c, y \rangle$ is inner product on \mathcal{V}
- variable $x \in \mathcal{V}$, $S \in \mathbf{S}^m$

dual SDP

$$\begin{array}{ll} \text{maximize} & -\text{Tr}(BZ) \\ \text{subject to} & \mathcal{A}^*(Z) + c = 0, \quad Z \succeq 0 \end{array}$$

- $\mathcal{A}^* : \mathbf{S}^m \rightarrow \mathcal{V}$ is adjoint of \mathcal{A}
- variable $Z \in \mathbf{S}^m$

Primal-dual path-following algorithm

(Tütüncü, Toh, Todd)

starting point: $S \succ 0$, $Z \succ 0$, any y

repeat:

1. *Verify stopping criteria.*
2. *Compute the Nesterov-Todd scaling matrix R : R is defined by*

$$R^T S^{-1} R = \mathbf{diag}(\lambda)^{-1}, \quad R^T Z R = \mathbf{diag}(\lambda), \quad \lambda \in \mathbf{R}_{++}^m$$

3. *Compute affine scaling directions:*

$$\begin{aligned} \mathcal{H}(\Delta Z^a S + Z \Delta S^a) &= -\mathbf{diag}(\lambda)^2 \\ \Delta S^a + \mathcal{A}(\Delta y^a) &= -(\mathcal{A}(y) + S - B) \\ \mathcal{A}^*(\Delta Z^a) &= -(\mathcal{A}^*(Z) + c) \end{aligned}$$

where $\mathcal{H}(X) = \frac{1}{2}(R^T X R^{-T} + R^{-1} X^T R)$

4. *Compute centering-corrector steps:*

$$\begin{aligned}\mathcal{H}(\Delta Z^c S + Z \Delta S^c) &= \rho I - \mathcal{H}(\Delta Z^a \Delta S^a) \\ \Delta S^c + \mathcal{A}(\Delta y^c) &= 0 \\ \mathcal{A}^*(\Delta Z^c) &= 0\end{aligned}$$

with ρ calculated based on $\mathbf{Tr}(SZ)$, ΔZ^a , ΔS^a

5. *Update primal and dual iterates:*

$$y := y + \alpha \Delta y, \quad S := S + \alpha \Delta S, \quad Z := Z + \beta \Delta Z$$

where $\Delta y = \Delta y^a + \Delta y^c$, $\Delta S = \Delta S^a + \Delta S^c$, $\Delta Z = \Delta Z^a + \Delta Z^c$, and

$$\begin{aligned}\alpha &= \min\{1, 0.99 \sup\{\alpha \mid S + \alpha \Delta S \succeq 0\}\} \\ \beta &= \min\{1, 0.99 \sup\{\beta \mid Z + \beta \Delta Z \succeq 0\}\}\end{aligned}$$

Overall complexity

- number of iterations is low (< 30), almost independent of problem size
- at each iteration, solve two sets of linear equations

$$\mathcal{H}(\Delta Z S + Z \Delta S) = D_1$$

$$\Delta S + \mathcal{A}(\Delta y) = D_2$$

$$\mathcal{A}^*(\Delta Z) = d$$

where

$$\mathcal{H}(X) = \frac{1}{2}(R^T X R^{-T} + R^{-1} X^T R)$$

values of R (NT scaling matrix), D_1 , D_2 , d change at each iteration

- search equations for other types of primal-dual methods are similar

General-purpose implementation

solve search equations as follows:

- eliminate ΔS :

$$\begin{aligned} -W\Delta ZW + \mathcal{A}(\Delta y) &= D \\ \mathcal{A}^*(\Delta Z) &= d \end{aligned}$$

where $W = RR^T$

- eliminate ΔZ :

$$\mathcal{A}^*(W^{-1}\mathcal{A}(\Delta y)W^{-1}) = d + \mathcal{A}^*(W^{-1}DW^{-1}) \quad (1)$$

a (usually dense) positive definite set of linear equations of Δy

overall cost: cost of forming coefficient matrix of (1) plus cost of solving

Search equation for KYP-SDPs

for simplicity, consider KYP-SDP with $N = 1$, $m_1 = 1$:

$$\begin{aligned} & \text{minimize} && c^T x + \mathbf{Tr}(CP) \\ & \text{subject to} && \mathcal{K}(P) + \mathcal{M}(x) \succeq N \end{aligned}$$

where

$$\mathcal{K}(P) = \begin{bmatrix} A^T P + PA & PB \\ B^T P & 0 \end{bmatrix}, \quad \mathcal{M}(x) = \sum_{i=1}^p x_i M_i$$

- $A \in \mathbf{R}^{n \times n}$, $B \in \mathbf{R}^{n \times 1}$
- (A, B) controllable; without loss of generality, can assume A is stable
- $p + n(n + 1)/2$ variables

search equations

$$W\Delta ZW + \mathcal{K}(\Delta P) + \mathcal{M}(\Delta x) = D_1$$

$$\mathcal{K}^*(\Delta Z) = D_2$$

$$\mathcal{M}^*(\Delta Z) = d$$

- $W \succ 0$; values of W , D_1 , D_2 , d change at each iteration
- \mathcal{K}^* , \mathcal{M}^* are adjoint mappings of \mathcal{K} , \mathcal{M} :

$$\mathcal{M}^*(\Delta Z) = (\mathbf{Tr}(M_1\Delta Z), \dots, \mathbf{Tr}(M_p\Delta Z))$$

$$\mathcal{K}^*(\Delta Z) = A\Delta Z_{11} + \Delta Z_{11}A^T + B\Delta\tilde{z}^T + \Delta\tilde{z}B^T$$

where

$$\Delta Z = \begin{bmatrix} \Delta Z_{11} & \Delta\tilde{z} \\ \Delta\tilde{z}^T & 2\Delta z_{n+1} \end{bmatrix}$$

Standard method of solving the search equations

$$W\Delta ZW + \mathcal{K}(\Delta P) + \mathcal{M}(\Delta x) = D_1$$

$$\mathcal{K}^*(\Delta Z) = D_2$$

$$\mathcal{M}^*(\Delta Z) = d$$

general-purpose solvers eliminate ΔZ from first equation:

$$\mathcal{K}^*(W^{-1}(\mathcal{K}(\Delta P) + \mathcal{M}(\Delta x))W^{-1}) = \mathcal{K}^*(W^{-1}D_1W^{-1}) - D_2$$

$$\mathcal{M}^*(W^{-1}(\mathcal{K}(\Delta P) + \mathcal{M}(\Delta x))W^{-1}) = \mathcal{M}^*(W^{-1}D_1W^{-1}) - d$$

a dense set of linear equations in $\Delta P, \Delta x$

cost: at least $O(n^6)$

Alternative method

step 1: express (1,1)-block of dual variable

$$\Delta Z = \begin{bmatrix} \Delta Z_{11} & \Delta \tilde{z} \\ \Delta \tilde{z}^T & 2\Delta z_{n+1} \end{bmatrix}$$

in terms of last column $\Delta z = (\Delta \tilde{z}, \Delta z_{n+1}) \in \mathbf{R}^{n+1}$, using 2nd equation

$$\mathcal{K}^*(\Delta Z) = A\Delta Z_{11} + \Delta Z_{11}A^T + B\Delta \tilde{z}^T + \Delta \tilde{z}B^T = D_2$$

this gives

$$\Delta Z_{11} = \sum_{i=1}^n \Delta z_i X_i - X_0$$

where X_0, \dots, X_n are defined by

$$AX_0 + X_0A^T + D_2 = 0, \quad AX_i + X_iA^T + Be_i^T + e_iB^T = 0, \quad i = 1, \dots, n$$

in other words,

$$\mathcal{K}^*(\Delta Z) = D_2 \iff \Delta Z = \mathcal{B}(\Delta z) - Z_0,$$

where

$$\mathcal{B}(\Delta z) = \begin{bmatrix} \sum_{i=1}^n \Delta z_i X_i & \Delta \tilde{z} \\ \Delta \tilde{z}^T & 2\Delta z_{n+1} \end{bmatrix}, \quad Z_0 = \begin{bmatrix} X_0 & 0 \\ 0 & 0 \end{bmatrix}$$

substituting in search equations gives

$$\begin{aligned} W\mathcal{B}(\Delta z)W + \mathcal{K}(\Delta P) + \mathcal{M}(\Delta x) &= D_1 + WZ_0W \\ \mathcal{M}^*(\mathcal{B}(\Delta z)) &= d \end{aligned}$$

variables $\Delta z \in \mathbf{R}^{n+1}$, $\Delta P \in \mathbf{S}^n$, $\Delta x \in \mathbf{R}^p$

step 2: eliminate ΔP from

$$\begin{aligned} W\mathcal{B}(\Delta z)W + \mathcal{K}(\Delta P) + \mathcal{M}(\Delta x) &= D_1 + WZ_0W \\ \mathcal{M}^*(\mathcal{B}(\Delta z)) &= d \end{aligned}$$

by noting that

$$S = \mathcal{K}(\Delta P) \text{ for some } \Delta P \iff \mathcal{B}^*(S) = 0$$

reduced equations:

$$\begin{aligned} \mathcal{B}^*(W\mathcal{B}(\Delta z)W) + \mathcal{B}^*(\mathcal{M}(\Delta x)) &= \mathcal{B}^*(D_1 + WZ_0W) \\ \mathcal{M}^*(\mathcal{B}(\Delta z)) &= d \end{aligned}$$

a set of $n + p + 1$ linear equations in $n + p + 1$ variables $\Delta z, \Delta x$

summary: reduced search equations

$$\begin{bmatrix} P_{11} & P_{12} \\ P_{12}^T & 0 \end{bmatrix} \begin{bmatrix} \Delta z \\ \Delta x \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}$$

- cost of solving is $O(n^3)$ operations (if we assume $p = O(n)$)
- from $\Delta z, \Delta x$, can find $\Delta Z, \Delta P$ in $O(n^3)$ operations
- P_{12} is independent of current iterates (W) and can be pre-computed

total cost: $O(n^3)$ plus the cost of constructing P_{11}

Constructing the reduced equations

$$P_{11} = \begin{bmatrix} H & 0 \\ 0 & 0 \end{bmatrix} + 2 \begin{bmatrix} W_{11} \\ W_{21} \end{bmatrix} \begin{bmatrix} G & 0 \end{bmatrix} + 2 \begin{bmatrix} G^T \\ 0 \end{bmatrix} \begin{bmatrix} W_{11} & W_{12} \end{bmatrix} \\ + 2W_{22}W + 2 \begin{bmatrix} W_{12} \\ W_{22} \end{bmatrix} \begin{bmatrix} W_{21} & W_{22} \end{bmatrix}$$

where

$$H_{ij} = \mathbf{Tr}(X_i W_{11} X_j W_{11}), \quad G = \begin{bmatrix} X_1 W_{12} & X_2 W_{12} & \cdots & X_n W_{12} \end{bmatrix}$$

and W is partitioned as $W = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix}$, $W_{11} \in \mathbf{S}^n$

cost dominated by $O(n^4)$ to precompute X_i 's; $O(n^4)$ to form H

Numerical example

$n = p$	KYP IPM		SeDuMi (primal)
	prep. time	time/iter.	time/iter.
25	0.1	0.07	0.1
50	1.2	0.3	7.4
100	21.7	3.3	324.7
200	438.3	31.6	

- CPU time in seconds on 2.4GHz PIV with 1GB of memory
- KYP-IPM: Matlab implementation of path-following method, using reduced search equations
- SeDuMi (primal): SeDuMi version 1.05 applied to primal problem
- prep. time is time to compute matrices X_i
- #iterations in both methods is comparable (7–15)

Reformulation of dual problem

primal SDP

$$\begin{aligned} & \text{minimize} && c^T x + \mathbf{Tr}(CP) \\ & \text{subject to} && \begin{bmatrix} A^T P + PA & PB \\ B^T P & 0 \end{bmatrix} + \sum_{i=1}^p x_i M_i \succeq N \end{aligned}$$

dual SDP

$$\begin{aligned} & \text{maximize} && -\mathbf{Tr}(NZ) \\ & \text{subject to} && AZ_{11} + Z_{11}A^T + \tilde{z}B^T + B\tilde{z}^T = N \\ & && \begin{bmatrix} Z_{11} & \tilde{z} \\ \tilde{z}^T & 2z_{n+1} \end{bmatrix} \succeq 0 \end{aligned}$$

can eliminate Z_{11} : $Z_{11} = -Z_0 + \sum_{i=1}^n z_i X_i$ where

$$AZ_0 + Z_0A^T + N = 0, \quad AX_i + X_iA^T + Be_i^T + e_iB^T = 0, \quad i = 1, \dots, n$$

results in an SDP with $n + 1$ variables, and an LMI of size $n + 1$

numerical example

$n = p$	KYP IPM		SeDuMi (dual)	
	prep. time	time/iter	prep. time	time/iter
25	0.1	0.07	0.1	0.02
50	1.2	0.3	1.2	0.2
100	21.7	3.3	22.5	3.0
200	438.3	31.6	436.2	26.5

- solution times are comparable
- preprocessing in both methods: computation of matrices X_i
- in fact both methods are equivalent: search equations for modified dual problem are equal to reduced search equations for primal problem

Fast construction of reduced system

use factorization of A to compute

$$H_{ij} = \mathbf{Tr}(X_i W_{11} X_j W_{11}), \quad i, j = 1, \dots, n$$

without computing X_i , *i.e.*, without explicitly solving

$$AX_i + X_i A^T + B e_i^T + e_i B^T = 0, \quad i = 1, \dots, n$$

- advantages: no need to store matrices X_i , faster construction of reduced search equations
- possible factorizations: eigenvalue decomposition, companion form, . . .

example: suppose A has distinct eigenvalues

$$A = V \mathbf{diag}(\lambda) V^{-1}$$

closed-form expression for H :

$$H = 2\text{Re} (V^{-T} (C \odot C^T) V^{-1} + V^{-*} (D \odot E^T) V^{-1})$$

where \odot is Hadamard product and

$$C = \Sigma \mathbf{diag}(V^{-1}B)^* V^* W_{11} V$$

$$D = V^* W_{11} V$$

$$E = \Sigma \mathbf{diag}(V^{-1}B)^* V^* W_{11} V \mathbf{diag}(V^{-1}B) \Sigma$$

$$\Sigma_{ij} = 1/(\lambda_i + \lambda_j^*), \quad i, j = 1, \dots, n$$

allows us to construct H in $O(n^3)$ operations

existence of distinct stable eigenvalues

- by assumption, (A, B) is controllable; hence can arbitrarily assign eigenvalues of $A + BK$ by choosing K
- choose $T = \begin{bmatrix} I & 0 \\ K & I \end{bmatrix}$, and replace LMI by equivalent LMI

$$T^T \left(\begin{bmatrix} A^T P + PA & PB \\ B^T P & 0 \end{bmatrix} + \sum_{i=1}^N x_i M_i \right) T \succeq T^T N T$$

$$\begin{bmatrix} (A + BK)^T P + P(A + BK) & PB \\ B^T P & 0 \end{bmatrix} + \sum_{i=1}^N x_i (T^T M_i T) \succeq T^T N T$$

conclusion: can assume without loss of generality that A is stable with distinct eigenvalues

Numerical example

five randomly generated problems with $p = 50$, $n = 100, \dots, 500$

n	KYP IPM		SeDuMi (dual)	
	prep. time	time/iter	prep. time	time/iter
100	1.3	1.2	0.7	1.4
200	10.1	8.9	3.8	23.2
300	32.4	27.3	11.9	127.3
400	72.2	62.0		
500	140.4	119.4		

- KYP-IPM is same algorithm as before, but uses eigenvalue decomposition of A to construct reduced search equations
- preprocessing time and time/iteration grow as $O(n^3)$

Conclusions

SDPs derived from the KYP-lemma

- a useful class of SDPs, widely encountered in control
- difficult to solve using general-purpose software

fast solution using interior-point methods

- reformulate dual SDP and solve using general-purpose solver (cost roughly $O(n^4)$)
- custom implementation based on fast solution of search equations (cost $O(n^4)$ or $O(n^3)$)