Applications of SDP least-squares in finance and combinatorics

Jérôme MALICK

CNRS, Lab. J. Kuntzmann, Grenoble

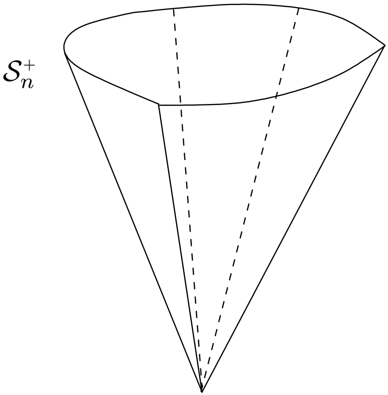
CORE math. prog. seminar – 11 March 2008

Outline

1. Semidefinite programming and least-squares
2. Appli 1: constructing covariance matrices in finance
3. Appli 2: relaxing binary quadratic problems
4. Appli 3: regularizing methods for solving SDP

Outline

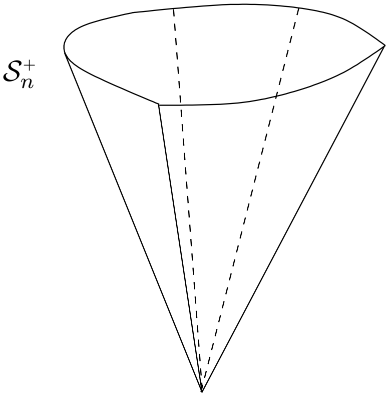
Semidefinite programming



The cone of positive semidef. matrices

$$\begin{aligned}\mathcal{S}_n^+ &= \{A \in \mathcal{S}_n, \ \ \forall x \in \mathbb{E}, \ x^\top A x \geqslant 0\} \\ &= \{A \in \mathcal{S}_n, \ \ \lambda_{\min}(A) \geqslant 0\}\end{aligned}$$

Semidefinite programming



The cone of positive semidef. matrices

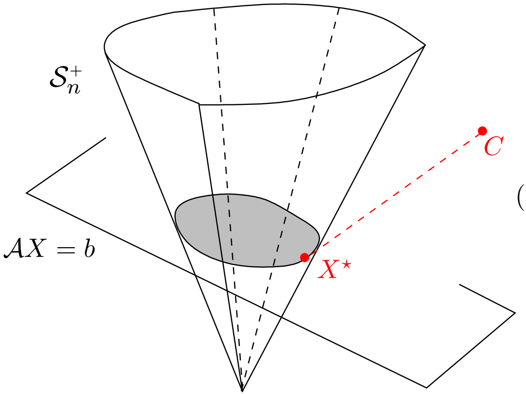
$$\begin{aligned} \mathcal{S}_n^+ &= \{A \in \mathcal{S}_n, \ \ \forall x \in \mathbb{E}, \ x^\top A x \geqslant 0\} \\ &= \{A \in \mathcal{S}_n, \ \ \lambda_{\min}(A) \geqslant 0\} \end{aligned}$$

Standard semidefinite programming

$$(\mathrm{SDP}) \left\{ \begin{array}{ll} \min & \langle C, X\rangle \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{array} \right.$$

R. Saigal, L. Vandenberghe, and H. Wolkowicz
Handbook of Semidefinite Programming
Kluwer, 2000

Semidefinite least-squares



$$\text{(SDLS)} \begin{cases} \min \quad \|X - C\|^2 \\ \quad \mathcal{A}X = b \\ \quad X \succcurlyeq 0 \end{cases}$$

SDLS problems

Example: compute the nearest correlation matrix

$$\left\{\begin{array}{l} \min \quad \|X - C\|^2 \\ \quad \operatorname{diag}(X) = e \\ \quad X \succcurlyeq 0 \end{array}\right.$$

SDLS problems

Example: compute the nearest correlation matrix

$$\begin{cases} \min & \|X - C\|^2 \\ & \operatorname{diag}(X) = e \\ & X \succcurlyeq 0 \end{cases}$$

In general: general form tackled

$$\begin{cases} \min & \langle C', X\rangle + \|W(X - C)W\|^2 \\ & \mathcal{A}X = b \\ & \mathcal{A}'X \leqslant b' \\ & X \succcurlyeq 0 \end{cases}$$

SDLS problems

Example: compute the nearest correlation matrix

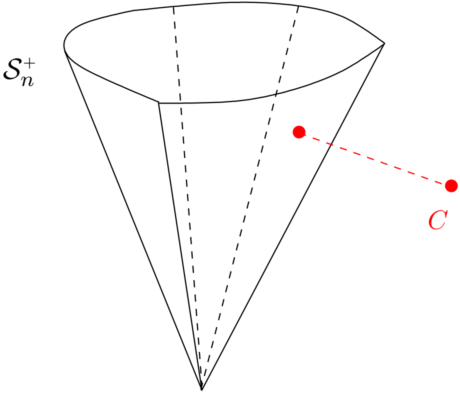
$$\begin{cases} \min & \|X - C\|^2 \\ & \operatorname{diag}(X) = e \\ & X \succcurlyeq 0 \end{cases}$$

In general: general form tackled

$$\begin{cases} \min & \langle C', X \rangle + \|W(X - C)W\|^2 \\ & \mathcal{A}X = b \\ & \mathcal{A}'X \leqslant b' \\ & X \succcurlyeq 0 \end{cases}$$

Question: How to solve semidefinite least-squares ?

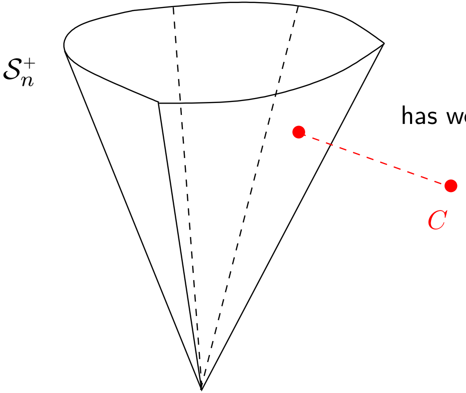
Basic fact: Project onto the cone



With no affine constraint, the problem is to project onto the cone

$$\begin{cases} \min & \|X - C\|^2 \\ & X \succcurlyeq 0 \end{cases}$$

Basic fact: Project onto the cone



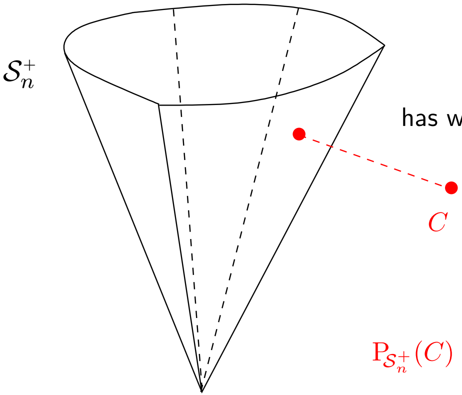
With no affine constraint, the problem is to project onto the cone

$$\begin{cases} \min & \|X - C\|^2 \\ & X \succcurlyeq 0 \end{cases}$$

has well-known easy-to-compute solution

$$C = P \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} P^\top$$

Basic fact: Project onto the cone



With no affine constraint, the problem is to project onto the cone

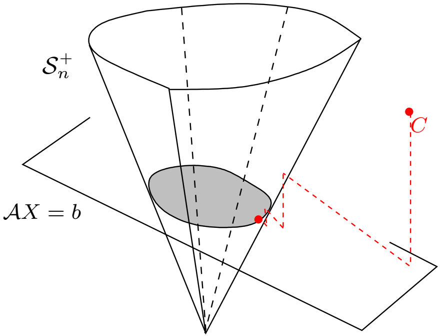
$$\left\{ \begin{array}{cc} \min & \|X - C\|^2 \\ X \succcurlyeq 0 & \end{array} \right.$$

has well-known easy-to-compute solution

$$C = P \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} P^\top$$

$$\mathrm{P}_{\mathcal{S}_n^+}(C) = P \begin{bmatrix} \max\{0, \lambda_1\} & & \\ & \ddots & \\ & & \max\{0, \lambda_n\} \end{bmatrix} P^\top$$

1st idea: alternating projections



Projection onto the intersection of two (convex) sets

$\longrightarrow$ alternating projections
$+$ Dykstra correction

N. Higham
Computing nearest symmetric correlation matrix - a problem from finance
IMA Journal of Numerical Analysis, 2002.

Next ideas

- 2nd idea: reformulate SDLS as a linear conic program

$$\text{(SDLS)} \iff \begin{cases} \min \quad t \\ \quad \|X - C\| \leqslant t \\ \quad \mathcal{A}X = b, \ X \succcurlyeq 0 \end{cases}$$

with usual conic solvers (Sedumi, SDPT3,...) $\longrightarrow$ no good results

Next ideas

- 2nd idea: reformulate SDLS as a linear conic program

$$(\mathrm{SDLS}) \iff \begin{cases} \min \quad t \\ \quad \|X - C\| \leqslant t \\ \quad \mathcal{A}X = b, \;\; X \succcurlyeq 0 \end{cases}$$

with usual conic solvers (Sedumi, SDPT3,...) $\longrightarrow$ no good results

- 3rd idea: interior-point method

K.C. Toh, R.H. Tütüncü, and M.J. Todd
Inexact primal-dual path-following algorithms for a special class of convex quadratic SDP and related problems
Pacific Journal of Optimization, 2006

Next ideas

- 2nd idea: reformulate SDLS as a linear conic program

$$(\text{SDLS}) \iff \begin{cases} \min \quad t \\ \quad \|X - C\| \leqslant t \\ \quad \mathcal{A}X = b, \;\; X \succcurlyeq 0 \end{cases}$$

with usual conic solvers (Sedumi, SDPT3,...) $\longrightarrow$ no good results

- 3rd idea: interior-point method

K.C. Toh, R.H. Tütüncü, and M.J. Todd
Inexact primal-dual path-following algorithms for a special class of convex quadratic SDP and related problems
Pacific Journal of Optimization, 2006

- 4th idea: by duality (Lagrangian duality + dual resolution)

J. Malick
A dual approach to semidefinite least-squares problems
SIAM Journal on Matrix Analysis and Applications, 2004

Lagrangian duality

Apply standard machinery:

$$L(X;\lambda) = \|X - C\|^2 - \lambda^\top(\mathcal{A}X - b)$$

$$\theta(\lambda) = \begin{cases} \min & L(X;\lambda) \\ X \succcurlyeq 0 \end{cases}$$

Lagrangian duality

Apply standard machinery:

$$L(X;\lambda) = \|X - C\|^2 - \lambda^\top(\mathcal{A}X - b)$$

$$\theta(\lambda) = \begin{cases} \min & L(X;\lambda) \\ X \succcurlyeq 0 \end{cases}$$

$$\text{(dual)} \begin{cases} \max & \theta(\lambda) \\ \lambda \in \mathbb{R}^m \end{cases}$$

dual problem is convex and differentiable !

Solving the dual problem

Apply standard algorithms:

$$(\text{dual}) \begin{cases} \max & \theta(\lambda) \\ & \lambda \in \mathbb{R}^m \end{cases}$$

variable metric descent

$$\lambda_{k+1} = \lambda_k - \tau_k W_k \nabla\theta(\lambda_k)$$

Solving the dual problem

Apply standard algorithms:

$$\text{(dual)} \begin{cases} \max & \theta(\lambda) \\ & \lambda \in \mathbb{R}^m \end{cases}$$

variable metric descent

$$\lambda_{k+1} = \lambda_k - \tau_k W_k \nabla\theta(\lambda_k)$$

1. Steepest descent = alternative projections

$$W_k = \mathrm{Id} \qquad \text{and} \qquad \tau_k = 1$$

Solving the dual problem

Apply standard algorithms:

$$(\text{dual}) \begin{cases} \max & \theta(\lambda) \\ & \lambda \in \mathbb{R}^m \end{cases}$$

variable metric descent

$$\lambda_{k+1} = \lambda_k - \tau_k W_k \nabla\theta(\lambda_k)$$

1. Steepest descent = alternative projections

$$W_k = \mathrm{Id} \quad \text{and} \quad \tau_k = 1$$

2. Quasi-Newton

$$W_k = \text{bfgs} \quad \text{and} \quad \tau_k \text{ well-chosen}$$

Solving the dual problem

Apply standard algorithms:

$$(\text{dual}) \begin{cases} \max & \theta(\lambda) \\ & \lambda \in \mathbb{R}^m \end{cases}$$

variable metric descent

$$\lambda_{k+1} = \lambda_k - \tau_k W_k \nabla\theta(\lambda_k)$$

1. Steepest descent = alternative projections

$$W_k = \mathrm{Id} \qquad \text{and} \qquad \tau_k = 1$$

2. Quasi-Newton

$$W_k = \text{bfgs} \qquad \text{and} \qquad \tau_k \text{ well-chosen}$$

3. Generalized Newton

$$W_k = [H_k]^{-1} \qquad \text{with } H_k \in \partial_c^2\theta(\lambda_k)$$

Numerical example

Compute the nearest correlation matrix (for random C of various sizes)

$$\begin{cases} \min \quad \|X - C\|^2 \\ \quad \operatorname{diag}(X) = e \\ \quad X \succcurlyeq 0 \end{cases}$$

Stop if relative error

$$\frac{\| \operatorname{diag}(X) - e\|}{\sqrt{n}} \leqslant 10^{-7}$$

Numerical example

Compute the nearest correlation matrix (for random C of various sizes)

$$\begin{cases} \min \quad \|X - C\|^2 \\ \quad \operatorname{diag}(X) = e \\ \quad X \succcurlyeq 0 \end{cases}$$

Stop if relative error

$$\frac{\|\operatorname{diag}(X) - e\|}{\sqrt{n}} \leqslant 10^{-7}$$

size	cpu time	nb iter.
100	0.2 s	14
500	16.3 s	17
1000	2 min 05 s	18
2000	17 min 41 s	19
3000	1h 08 min 14 s	19

Conclusion: SDP vs. SDLS

- Semidefinite programming

$$(\text{SDP}) \left\{ \begin{array}{ll} \min & \langle C, X \rangle \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{array} \right. \qquad \left\{ \begin{array}{ll} \min & b^\top \lambda \\ & \mathcal{A}^*\lambda - C = Z \\ & Z \succcurlyeq 0 \end{array} \right.$$

$$(\text{SDLS}) \left\{ \begin{array}{ll} \min & \|X - C\|^2 \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{array} \right. \qquad \left\{ \begin{array}{ll} \max & \theta(\lambda) \\ & \lambda \in \mathbb{R}^m \end{array} \right.$$

Conclusion: SDP vs. SDLS

- Semidefinite programming

$$
\text{(SDP)} \begin{cases} \min & \langle C, X\rangle \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{cases} \qquad \begin{cases} \min & b^\top \lambda \\ & \mathcal{A}^*\lambda - C = Z \\ & Z \succcurlyeq 0 \end{cases}
$$

$$
\text{(SDLS)} \begin{cases} \min & \|X - C\|^2 \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{cases} \qquad \begin{cases} \max & \theta(\lambda) \\ & \lambda \in \mathbb{R}^m \end{cases}
$$

- Bottom-line:

$\longrightarrow$ SDLS are easier than SDP !

(geometry + dual problem differentiable)

Conclusion: SDP vs. SDLS

- Semidefinite programming

$$(\text{SDP}) \left\{ \begin{array}{ll} \min & \langle C, X\rangle \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{array} \right. \qquad \left\{ \begin{array}{ll} \min & b^\top \lambda \\ & \mathcal{A}^*\lambda - C = Z \\ & Z \succcurlyeq 0 \end{array} \right.$$

$$(\text{SDLS}) \left\{ \begin{array}{ll} \min & \|X - C\|^2 \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{array} \right. \qquad \left\{ \begin{array}{ll} \max & \theta(\lambda) \\ & \lambda \in \mathbb{R}^m \end{array} \right.$$

- Bottom-line:

  $\longrightarrow$ SDLS are easier than SDP !

  (geometry + dual problem differentiable)

- SDLS modelling ? much less than SDP, but

  $\longrightarrow$ in next applications, SDLS problems naturally appear...

Outline

Estimating covariance matrices

- Random variable $X \in \mathbb{R}^n$ with unknown moments
  (Eg: returns of n assets)
- Observations of X : $x_t \in \mathbb{R}^n$ for $t = 1, \ldots, T$
  (Eg: historical charts of the assets)
- Empirical estimations

$$\mu = \frac{1}{T}\sum_{t=1}^{T} x_t \qquad \widetilde{\Gamma} = \frac{1}{T}\sum_{t=1}^{T}(x_t - \mu)(x_t - \mu)^\top \succcurlyeq 0$$

- Beyond statistics:

$\longrightarrow$ in practice, $\widetilde{\Gamma}$ has weaknesses
(Eg: incomplete data, choice of T, bad conditioning,...)

$\longrightarrow$ how to add exogenous (nonhistorical) information ?
(Eg: set values to entries, impose structure...)

Covariance matrices in finance

Goal: to construct "good" covariance matrices with special structure, from the empirical estimation $\widetilde{\Gamma}$

Many uses in quantitative finance, eg:

- Portfolio selection
- Correlation Stress testing
- Convexification of BGM model
- Aggregation to a global risk model from local models and cross-market estimates

RaisePartner Corp.

http://www.raisepartner.com

Example: portfolio selection

Find the portfolio $x = (x_1, \ldots, x_n)$ with minimum risk, as the solution of

$$\begin{cases} \min \quad x^\top \Gamma x \\ \quad x^\top \rho \geqslant r_0 \\ \quad x \in \Delta \end{cases}$$

Example: portfolio selection

Find the portfolio $x = (x_1, \ldots, x_n)$ with minimum risk, as the solution of

$$\begin{cases} \min \quad x^\top \Gamma x \\ \quad x^\top \rho \geqslant r_0 \\ \quad x \in \Delta \end{cases}$$

$\widetilde{\Gamma}$ ($\approx \Gamma$) underestimates small risks

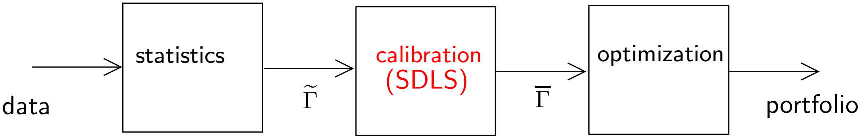
$\longrightarrow$ decision not reliable, portfolio unstable

Example: portfolio selection

Find the portfolio $x = (x_1, \ldots, x_n)$ with minimum risk, as the solution of

$$\begin{cases} \min \quad x^\top \Gamma x \\ \quad x^\top \rho \geqslant r_0 \\ \quad x \in \Delta \end{cases}$$

$\widetilde{\Gamma}$ $(\approx \Gamma)$ underestimates small risks
$\longrightarrow$ decision not reliable, portfolio unstable

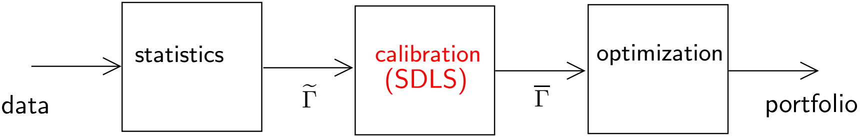


Example: portfolio selection

Find the portfolio $x = (x_1, \ldots, x_n)$ with minimum risk, as the solution of

$$\begin{cases} \min \quad x^\top \Gamma x \\ \quad x^\top \rho \geqslant r_0 \\ \quad x \in \Delta \end{cases}$$

$\widetilde{\Gamma}$ ($\approx \Gamma$) underestimates small risks
$\longrightarrow$ decision not reliable, portfolio unstable

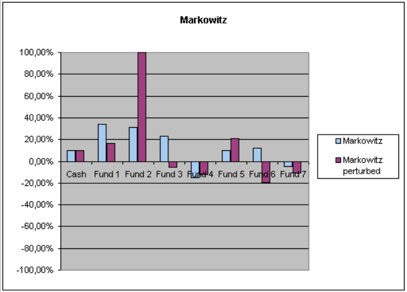


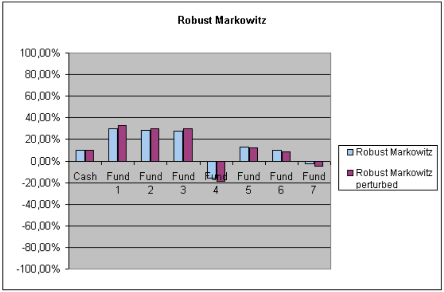
Preliminary calibration: in selection problem, use $\bar{\Gamma}$ the solution of

$$\begin{cases} \min \quad \|X - \widetilde{\Gamma}\|^2 \\ \quad X - \alpha I_n \succcurlyeq 0 \\ \quad \operatorname{trace}(X) = \operatorname{trace}(\widetilde{\Gamma}) \\ \quad \langle A_i, X \rangle \in [a_i, b_i] \end{cases}$$

Simple illustration

	Cash	Fund 1	Fund 2	Fund 3	Fund 4	Fund 5	Fund 6	Fund 7	Total
Markowitz	10,00%	33,79%	30,81%	23,32%	-14,92%	9,66%	12,13%	-4,78%	100,00%
Markowitz Perturbed	10,00%	16,61%	100,00%	-5,33%	-11,97%	21,15%	-19,61%	-10,85%	100,00%
Robust Markowitz	10,00%	30,05%	28,34%	27,73%	-16,21%	12,90%	10,10%	-2,91%	100,00%
Robust Markowitz Perturbe	10,00%	32,72%	29,97%	29,91%	-18,55%	12,39%	8,35%	-4,79%	100,00%





Outline

Bounds, relaxations

Bounding is a basic tool of combinatorial optimization
If we can't

$$\begin{cases} \max & \text{(easy objective function)} \\ x \in & \text{(difficult constraints)} \end{cases}$$

we are content with an upper bound, computing after relaxation

Bounds, relaxations

Bounding is a basic tool of combinatorial optimization
If we can't

$$\begin{cases} \max & \text{(easy objective function)} \\ x \in & \text{(difficult constraints)} \end{cases}$$

we are content with an upper bound, computing after relaxation

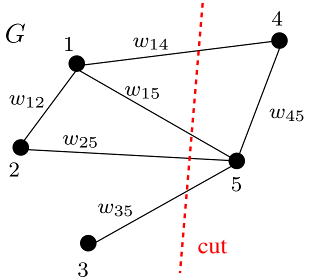
Enlarging the feasible set $\longrightarrow$ $\begin{cases} \max & \text{(easy fonction)} \\ x \in & \text{(easy constraints)} \end{cases}$

Example: max-cut

$G = (V, E)$ undirected graph
$Q = (-w_{ij})_{ij}$ weight-matrix

finding the cut of maximal weight

$$= \begin{cases} \max & x^\top Q\, x \\ & x \in \{-1,1\}^n \end{cases}$$



$\longrightarrow$ NP hard

$\longrightarrow$ Classical relaxations

- linear relaxation: $x_i \in \{-1,1\} \rightarrow x_i \in [-1,1]$
- SDP relaxation...
- many others !

SDP relaxation

$$(\text{max-cut}) \left\{ \begin{array}{ll} \max & x^\top Q\, x \\ & x \in \{-1,1\}^n \end{array} \right.$$

$$\boxed{X = xx^\top}$$

$$(\text{max-cut}) \left\{ \begin{array}{ll} \max & \langle Q, X\rangle \\ & X_{ii} = 1 \\ & X \succcurlyeq 0 \\ & \operatorname{rang} X = 1 \end{array} \right.$$

L. Lovász
On the Shannon capacity of a graph
IEEE Trans. of Information Theory, 1979

M. X. Goemans and D. P. Williamson
Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming
Journal of ACM, 1995

SDP relaxation

$$(\text{max-cut}) \begin{cases} \max & x^\top Q\, x \\ & x \in \{-1,1\}^n \end{cases}$$

L. Lovász
On the Shannon capacity of a graph
IEEE Trans. of Information Theory, 1979

M. X. Goemans and D. P. Williamson
Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming
Journal of ACM, 1995

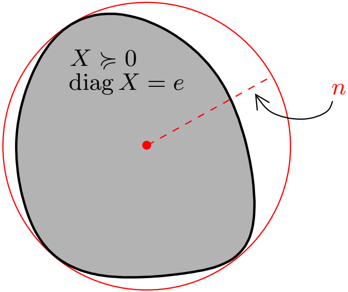
$$\boxed{X = xx^\top}$$

$$(\text{max-cut}) \begin{cases} \max & \langle Q, X\rangle \\ & X_{ii} = 1 \\ & X \succcurlyeq 0 \\ & \operatorname{rang} X = 1 \end{cases} \qquad (\text{SDP relax.}) \begin{cases} \max & \langle Q, X\rangle \\ & X_{ii} = 1 \\ & X \succcurlyeq 0 \end{cases}$$

Spherical constraint

For all $X \succcurlyeq 0$ satisfying $X_{ii} = 1$, we have

$$\|X\| \leqslant n$$



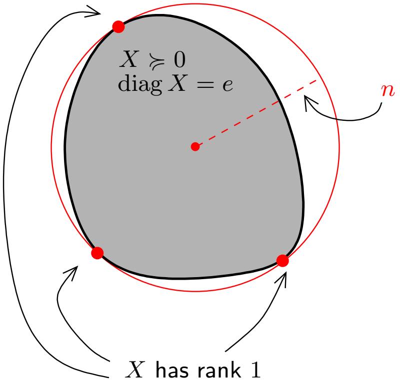
Spherical constraint

For all $X \succcurlyeq 0$ satisfying $X_{ii} = 1$, we have

$$\|X\| \leqslant n$$

$$\|X\| = n \iff \operatorname{rang} X = 1$$

"spherical constraint"



Dualizing spherical constraint

$$(\text{max-cut}) \begin{cases} \max & \langle Q, X\rangle \\ & X_{ii} = 1 \quad \text{for all } i \\ & X \succcurlyeq 0 \\ & \|X\|^2 - n^2 = 0 \end{cases}$$

Dualizing spherical constraint

$$(\text{max-cut}) \begin{cases} \max & \langle Q, X\rangle \\ & X_{ii} = 1 \quad \text{for all } i \\ & X \succcurlyeq 0 \\ & \|X\|^2 - n^2 = 0 \end{cases}$$

$$L(X;\alpha) = \langle Q, X\rangle - \alpha(\|X\|^2 - n^2)$$ Lagrangian

Dualizing spherical constraint

$$\text{(max-cut)} \begin{cases} \max & \langle Q, X \rangle \\ & X_{ii} = 1 \quad \text{for all } i \\ & X \succcurlyeq 0 \\ & \|X\|^2 - n^2 = 0 \end{cases}$$

$$L(X; \alpha) = \langle Q, X \rangle - \alpha(\|X\|^2 - n^2)$$

Lagrangian

$$\theta(\alpha) = \begin{cases} \max & L(X; \alpha) \\ & X_{ii} = 1 \quad \text{for all } i \\ & X \succcurlyeq 0 \end{cases}$$

dual function

Dualizing spherical constraint

$$(\text{max-cut}) \begin{cases} \max & \langle Q, X\rangle \\ & X_{ii} = 1 \quad \text{for all } i \\ & X \succcurlyeq 0 \\ & \|X\|^2 - n^2 = 0 \end{cases}$$

$$L(X;\alpha) = \langle Q, X\rangle - \alpha(\|X\|^2 - n^2)$$

Lagrangian

$$\theta(\alpha) \quad = \quad \begin{cases} \max & L(X;\alpha) \\ & X_{ii} = 1 \quad \text{for all } i \\ & X \succcurlyeq 0 \end{cases}$$

dual function

$$\theta(\alpha) = \text{Cste}_\alpha - \alpha \begin{cases} \min & \frac{1}{2}\|X - Q/\alpha\|^2 \\ & X_{ii} = 1 \quad \text{for all } i \\ & X \succcurlyeq 0 \end{cases}$$

$\longrightarrow$ SDLS (if $\alpha > 0$)

Bounds and duality gap

What does it yield ?

$\longrightarrow$ Weak duality: each $\theta(\alpha)$ gives an upper bound for (max-cut)

Bounds and duality gap

What does it yield ?

$\longrightarrow$ Weak duality: each $\theta(\alpha)$ gives an upper bound for (max-cut)

$\longrightarrow$ No gap (!): in theory, bounds as tight as we want !?

$$\theta(\alpha) \longrightarrow \mathrm{val}(\text{max-cut}) \qquad \text{when } \alpha \to -\infty$$

$\longrightarrow$ In practice:

Bounds and duality gap

What does it yield ?

$\longrightarrow$ Weak duality: each $\theta(\alpha)$ gives an upper bound for (max-cut)

$\longrightarrow$ No gap (!): in theory, bounds as tight as we want !?

$$\theta(\alpha) \longrightarrow \mathrm{val}(\text{max-cut}) \qquad \text{when } \alpha \to -\infty$$

$\longrightarrow$ In practice:

- if $\alpha > 0$, compute $\theta(\alpha)$ with SDLS solvers

Bounds and duality gap

What does it yield ?

$\longrightarrow$ Weak duality: each $\theta(\alpha)$ gives an upper bound for (max-cut)

$\longrightarrow$ No gap (!): in theory, bounds as tight as we want !?

$$\theta(\alpha) \longrightarrow \mathrm{val(max\text{-}cut)} \qquad \text{when } \alpha \to -\infty$$

$\longrightarrow$ In practice:

- if $\alpha > 0$, compute $\theta(\alpha)$ with SDLS solvers
- if $\alpha = 0$, there holds

$$\theta(0) = \begin{cases} \max & \langle Q, X \rangle \\ & X_{ii} = 1 \quad \text{for all } i \\ & X \succcurlyeq 0 \end{cases}$$

so compute $\theta(0)$ with SDP solvers

Bounds and duality gap

What does it yield ?

$\longrightarrow$ Weak duality: each $\theta(\alpha)$ gives an upper bound for (max-cut)

$\longrightarrow$ No gap (!): in theory, bounds as tight as we want !?

$$\theta(\alpha) \longrightarrow \mathrm{val}(\text{max-cut}) \qquad \text{when } \alpha \to -\infty$$

$\longrightarrow$ In practice:

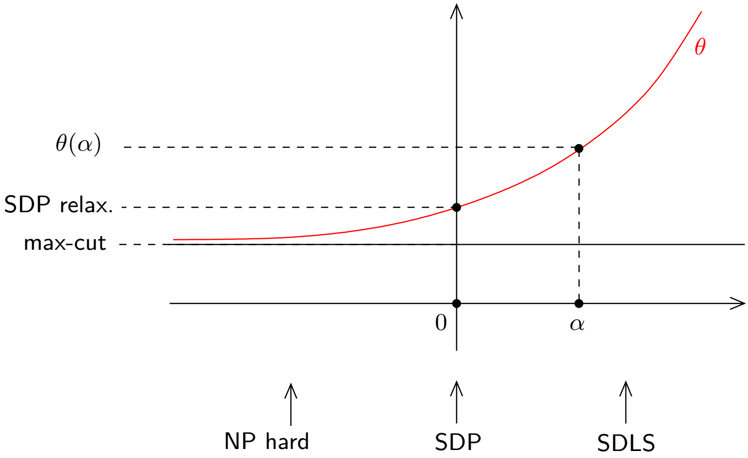
- if $\alpha > 0$, compute $\theta(\alpha)$ with SDLS solvers
- if $\alpha = 0$, there holds

$$\theta(0) = \begin{cases} \max & \langle Q, X \rangle \\ & X_{ii} = 1 \quad \text{for all } i \\ & X \succcurlyeq 0 \end{cases}$$

  so compute $\theta(0)$ with SDP solvers

- if $\alpha < 0$ large, computing $\theta(\alpha)$ is NP hard !

No duality gap



Numerical example

For a random max-cut (500 nodes and 70% density) compute SDLS bounds $\theta(\alpha)$ for several $\alpha \in [0, 1]$

α	error	iters	time
1	94%	9	28 s
0.1	52%	16	53 s
0.01	10%	33	106 s
0.001	1%	89	275 s

with a guarantee $$\frac{\theta(\alpha) - \theta(0)}{\theta(\alpha)} \leqslant \text{error}$$

Numerical example

For a random max-cut (500 nodes and 70% density) compute SDLS bounds $\theta(\alpha)$ for several $\alpha \in [0, 1]$

α	error	iters	time
1	94%	9	28 s
0.1	52%	16	53 s
0.01	10%	33	106 s
0.001	1%	89	275 s

with a guarantee $$\frac{\theta(\alpha) - \theta(0)}{\theta(\alpha)} \leqslant \text{error}$$

$\longrightarrow$ when $\alpha \downarrow 0$:

SDLS bounds get tighter (but always less than SDP one !)

SDLS problems become more difficult

Conclusion: SDLS bounds

Spherical constraint:

$\longrightarrow$ new formulation of rank-one constraint

$\longrightarrow$ new interpretation of SDP relaxation

$\longrightarrow$ opens the way for SDLS solvers

$\longrightarrow$ for any binary quadratic problem

J. Malick
Spherical constraint in Boolean quadratic problems
Journal of Global Optimization, 2007

Conclusion: SDLS bounds

Spherical constraint:

⟶ new formulation of rank-one constraint

⟶ new interpretation of SDP relaxation

⟶ opens the way for SDLS solvers

⟶ for any binary quadratic problem

J. Malick
Spherical constraint in Boolean quadratic problems
Journal of Global Optimization, 2007

Current research:

⟶ more complete numerical experiments

⟶ good balance between accuracy and speed of computation ?

⟶ difficult combinatorial problems (k-cluster, QAP)

Outline

How to solve general SDP

1. primal-dual interior point methods
2. spectral bundle methods, $\lambda_{\min}(X)$
3. with low-rank methods, $X = RR^\top$ with $R \in \mathbb{R}^{n \times r}$
4. and others ! Sorry for not citing them...

How to solve general SDP

1. primal-dual interior point methods
2. spectral bundle methods, $\lambda_{\min}(X)$
3. with low-rank methods, $X = RR^\top$ with $R \in \mathbb{R}^{n\times r}$
4. and others ! Sorry for not citing them...
5. but still need for something else ! Real-life applications (eg: in combinatorics) ask for bigger sizes or quicker solvers

   $\longrightarrow$ Idea: (primal) proximal algorithm

How to solve general SDP

1. primal-dual interior point methods
2. spectral bundle methods, $\lambda_{\min}(X)$
3. with low-rank methods, $X = RR^\top$ with $R \in \mathbb{R}^{n\times r}$
4. and others ! Sorry for not citing them...
5. but still need for something else ! Real-life applications (eg: in combinatorics) ask for bigger sizes or quicker solvers

$\longrightarrow$ Idea: (primal) proximal algorithm

$\longrightarrow$ Target problems: n not too big, m very large

$$n \leqslant 1000, \qquad m \text{ up to } 100,000$$

How to solve general SDP

1. primal-dual interior point methods
2. spectral bundle methods, $\lambda_{\min}(X)$
3. with low-rank methods, $X = RR^\top$ with $R \in \mathbb{R}^{n \times r}$
4. and others ! Sorry for not citing them...
5. but still need for something else ! Real-life applications (eg: in combinatorics) ask for bigger sizes or quicker solvers

$\longrightarrow$ Idea: (primal) proximal algorithm

$\longrightarrow$ Target problems: n not too big, m very large

$$n \leqslant 1000, \qquad m \text{ up to } 100,000$$

$\longrightarrow$ Joint work F. Rendl, J. Povh and A. Wiegele

Proximal approach on paper

$$\begin{cases} \min & \langle C, X \rangle \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{cases}$$

Proximal approach on paper

$$\left\{\begin{array}{ll} \min & \langle C, X\rangle \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{array}\right. \quad \Longleftrightarrow \quad \left\{\begin{array}{ll} \min & \langle C, X\rangle + \dfrac{1}{2\,t}\|X - Y\|^2 \\ & Y \in \mathcal{S}_n \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{array}\right.$$

Proximal approach on paper

$$\left\{\begin{array}{l} \min \quad \langle C, X\rangle \\ \quad \mathcal{A}X = b \\ \quad X \succcurlyeq 0 \end{array}\right. \iff \left\{\begin{array}{l} \min \quad \langle C, X\rangle + \frac{1}{2\,t}\|X - Y\|^2 \\ \quad Y \in \mathcal{S}_n \\ \quad \mathcal{A}X = b \\ \quad X \succcurlyeq 0 \end{array}\right.$$

Introduce the convex and differentiable (!) function

$$F_t(Y) = \left\{\begin{array}{l} \min \quad \langle C, X\rangle + \frac{1}{2\,t}\|X - Y\|^2 \\ \quad \mathcal{A}X = b \\ \quad X \succcurlyeq 0 \end{array}\right.$$

Proximal approach on paper

$$\left\{\begin{array}{ll}\min & \langle C, X\rangle \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0\end{array}\right. \quad \Longleftrightarrow \quad \left\{\begin{array}{ll}\min & \langle C, X\rangle + \frac{1}{2\,t}\|X - Y\|^2 \\ & Y \in \mathcal{S}_n \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0\end{array}\right.$$

Introduce the convex and differentiable (!) function

$$F_t(Y) = \left\{\begin{array}{ll}\min & \langle C, X\rangle + \frac{1}{2\,t}\|X - Y\|^2 \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0\end{array}\right.$$

Hence

$$\left\{\begin{array}{ll}\min & \langle C, X\rangle \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0\end{array}\right. \quad \Longleftrightarrow \quad \left\{\begin{array}{ll}\min & F_t(Y) \\ & Y \in \mathcal{S}_n\end{array}\right.$$

Proximal point in practice

General fact:

$$\begin{array}{ll} & \text{gradient step on } F_t \\ \iff & \text{fixed point iteration on prox point} \\ \iff & Y_{k+1} = \operatorname{prox}(Y_k) \end{array}$$

where

$$\operatorname{prox}(Y) = \begin{cases} \operatorname{argmin} & \langle C, X\rangle + \frac{1}{2\,t}\|X - Y\|^2 \\ \quad \mathcal{A}X = b \\ \quad X \succcurlyeq 0 \end{cases}$$

Proximal point in practice

General fact:

$$\begin{array}{ll} & \text{gradient step on } F_t \\ \iff & \text{fixed point iteration on prox point} \\ \iff & Y_{k+1} = \mathrm{prox}(Y_k) \end{array}$$

where

$$\mathrm{prox}(Y) = \begin{cases} \text{argmin} & \langle C, X\rangle + \frac{1}{2\,t}\|X - Y\|^2 \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{cases}$$

Practical implementation requires

- an algorithm to compute $\mathrm{prox}(Y_k) \longrightarrow$ SDLS
- a rule to stop this algorithm (in practice $Y_{k+1} \approx \mathrm{prox}(Y_k)$)

Solving the outer problem

Algorithm (General proximal algorithm for SDP)

Repeat until $\|Z + \mathcal{A}^* y - C\|$ *small:*

Repeat until $\|b - \mathcal{A}X\|$ *small enough:*

Compute $X = t(\mathcal{A}^* y - C + Y/t)_+$, $Z = -(\mathcal{A}^* y - C + Y/t)_-$

Set $g = b - \mathcal{A}X$

Update $y \leftarrow y + \tau W g$ *with appropriate* τ *and* W

end (inner repeat)

$Y \leftarrow X$

end (outer repeat)

Solving the outer problem

Algorithm (General proximal algorithm for SDP)

Repeat until $\|Z + \mathcal{A}^* y - C\|$ *small:*
Repeat until $\|b - \mathcal{A}X\|$ *small enough:*
Compute $X = t(\mathcal{A}^* y - C + Y/t)_+$, $Z = -(\mathcal{A}^* y - C + Y/t)_-$
Set $g = b - \mathcal{A}X$
Update $y \leftarrow y + \tau W g$ *with appropriate* τ *and* W
end (inner repeat)
$Y \leftarrow X$
end (outer repeat)

- Under some technical assumptions:

  any accumulation point of generated sequence is solution of SDP

Solving the outer problem

Algorithm (General proximal algorithm for SDP)

Repeat until $\|Z + \mathcal{A}^* y - C\|$ *small:*
  Repeat until $\|b - \mathcal{A}X\|$ *small enough:*
    Compute $X = t(\mathcal{A}^* y - C + Y/t)_+$, $Z = -(\mathcal{A}^* y - C + Y/t)_-$
    Set $g = b - \mathcal{A}X$
    Update $y \leftarrow y + \tau W g$ *with appropriate* τ *and* W
  end (inner repeat)
  $Y \leftarrow X$
end (outer repeat)

- Under some technical assumptions:

  any accumulation point of generated sequence is solution of SDP

- Connexion with boundary points: If $\mathcal{A}$ is surjective,

$$W = [\mathcal{A}\mathcal{A}^*]^{-1} \qquad \tau = 1/t$$

  then: the present algorithm = the boundary point method

J. Malick, J. Povh, F. Rendl and A. Wiegele
Boundary point method to solve semidefinite programs
Computing, 2006

In practice

regularization algorithms:

$$\text{projections methods} \perp \text{interior points}$$

Complementary and semidefiniteness are ensured throughout, while primal dual feasibility are obtained asymptotically

In practice

regularization algorithms:

projections methods $\perp$ interior points

Complementary and semidefiniteness are ensured throughout, while primal dual feasibility are obtained asymptotically

After testing various prototypes, we chose: one single inner iteration, $W = [\mathcal{A}\mathcal{A}^*]^{-1}$, $\tau = 1/t$, update strategy of parameters...

Algorithm (Final algorithm)

Repeat until $\delta <$*tolerance:*
Compute the solution y *of* $\mathcal{A}\mathcal{A}^* y = \mathcal{A}(C - Z) + (b - \mathcal{A}Y)/t$
Compute $S = Y/t + \mathcal{A}^* y - C$, $Z = -S_-$ *and* $X = S_+$
Update $Y \leftarrow X$, *compute* δ *and update* t

In practice

regularization algorithms:

projections methods $\perp$ interior points

Complementary and semidefiniteness are ensured throughout, while primal dual feasibility are obtained asymptotically

After testing various prototypes, we chose: one single inner iteration, $W = [\mathcal{A}\mathcal{A}^*]^{-1}$, $\tau = 1/t$, update strategy of parameters...

Algorithm (Final algorithm)

Repeat until $\delta <$*tolerance:*
Compute the solution y *of* $\mathcal{A}\mathcal{A}^* y = \mathcal{A}(C - Z) + (b - \mathcal{A}Y)/t$
Compute $S = Y/t + \mathcal{A}^* y - C$, $Z = -S_-$ *and* $X = S_+$
Update $Y \leftarrow X$, *compute* δ *and update* t

$\mathcal{A}\mathcal{A}^*$ (and its Choleski factorization) computed one single time !

Numerical experiments

$\longrightarrow$ 1st test: Random instances with large m

(Generator is online, so data reproducible)

`http//bipop.inrialpes.fr/people/malick/software`

$$(\text{SDP}) \left\{ \begin{array}{ll} \min & \langle C, X \rangle \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{array} \right. \qquad (\text{dual}) \left\{ \begin{array}{ll} \max & b^\top y \\ & C - \mathcal{A}^* y = Z \\ & Z \succcurlyeq 0 \end{array} \right.$$

Numerical experiments

⟶ 1st test: Random instances with large m

(Generator is online, so data reproducible)

`http//bipop.inrialpes.fr/people/malick/software`

$$(\text{SDP}) \begin{cases} \min & \langle C, X\rangle \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{cases} \qquad (\text{dual}) \begin{cases} \max & b^\top y \\ & C - \mathcal{A}^* y = Z \\ & Z \succcurlyeq 0 \end{cases}$$

⟶ Comparison with 2 strong codes - on a standard machine

K.C. Toh
Solving large scale semidefinite programs via an iterative solver on the augmented systems
SIAM Journal on Optimization, 2004.

S. Burer and R.D.C Monteiro
A nonlinear programming algorithm for solving SDP via low-rank factorization
Mathematical Programming, 2003

Random instances

Stopping tolerance for our regularization method:

$$\text{error} := \max\left\{\frac{\|\mathcal{A}X - b\|}{1+\|b\|}, \frac{\|C - Z - \mathcal{A}^*y\|}{1+\|C\|}\right\} \leqslant 10^{-7}$$

n	m	prox time
400	30000	153
500	30000	202
500	40000	217
700	50000	592
700	70000	534

Random instances

Stopping tolerance for our regularization method:

$$\text{error} := \max\left\{\frac{\|\mathcal{A}X - b\|}{1+\|b\|}, \frac{\|C - Z - \mathcal{A}^* y\|}{1+\|C\|}\right\} \leqslant 10^{-7}$$

		prox	Toh		SDPLR	
n	m	time	time	error	time	error
400	30000	153	1000	0.6e-7	1880	7.3e-6
500	30000	202	1309	0.5e-7	2165	7.8e-6
500	40000	217	1668	0.4e-7	3600	8.8e-6
700	50000	592	2696	0.8e-7	3600	3.4e-5
700	70000	534	4065	0.3e-7	3600	6.3e-5

Random instances

Stopping tolerance for our regularization method:

$$\text{error} := \max \left\{ \frac{\|\mathcal{A}X - b\|}{1 + \|b\|}, \frac{\|C - Z - \mathcal{A}^* y\|}{1 + \|C\|} \right\} \leqslant 10^{-7}$$

		prox	Toh		SDPLR	
n	m	time	time	error	time	error
400	30000	153	1000	0.6e-7	1880	7.3e-6
500	30000	202	1309	0.5e-7	2165	7.8e-6
500	40000	217	1668	0.4e-7	3600	8.8e-6
700	50000	592	2696	0.8e-7	3600	3.4e-5
700	70000	534	4065	0.3e-7	3600	6.3e-5

Note: If the factorization given, larger instances ($n = 1000$, $m = 150,000$) in less than one hour

Example in combinatorics: Lovász theta number

Let G be a graph (and $\overline{G}$ its complementary)

$$\textcolor{red}{\vartheta}(G) := \begin{cases} \max \quad \langle \mathbf{1}_{n\times n}, X\rangle \\ \quad X_{ij} = 0, \quad \forall\, i,j \in E(G) \\ \quad \operatorname{trace} X = 1 \\ \quad X \succcurlyeq 0 \end{cases}$$

It holds
$$\omega(G) \leqslant \vartheta(\overline{G}) \leqslant \chi(G)$$

The clique number $\omega(G)$ and the chromatic number $\chi(G)$ are "impossible" to compute and even hard to approximate

L. Lovász
On the Shannon capacity of a graph
IEEE Trans. Inform. Theory, 1979

Computing Lovász theta number

- Challenge: Lovász number of certain graphs of the DIMACS collection not computed so far...

Computing Lovász theta number

- Challenge: Lovász number of certain graphs of the DIMACS collection not computed so far...
- Big help: for Lovasz theta number, $\mathcal{A}\mathcal{A}^*$ is diagonal !

Computing Lovász theta number

- Challenge: Lovász number of certain graphs of the DIMACS collection not computed so far...
- Big help: for Lovasz theta number, $\mathcal{A}\mathcal{A}^*$ is diagonal !
- Results: from DIMACS library

graph name	$\lvert V(G)\rvert$	$\lvert E(\overline{G})\rvert$	$\vartheta(\overline{G})$	$\lvert E(G)\rvert$	$\vartheta(G)$
brock400-1	400	20077	39.702	59723	10.388
keller5	776	74710	31.000	225990	31.000
brock800-1	800	112095	42.222	207505	19.233
p-hat500-1	500	93181	13.074	31569	58.036
p-hat1000-3	1000	127754	84.80	371746	18.23

Conclusion: regularization method

Our new method for solving SDP

$\longrightarrow$ follows a very general approach of convex analysis

$\longrightarrow$ takes an opposite direction to Interior Points

$\longrightarrow$ compares favorably with other solvers for some problems

J. Malick, J. Povh, F. Rendl and A. Wiegele
Regularization methods for semidefinite programming
Submitted, 2007

Conclusion: regularization method

Our new method for solving SDP

$\longrightarrow$ follows a very general approach of convex analysis

$\longrightarrow$ takes an opposite direction to Interior Points

$\longrightarrow$ compares favorably with other solvers for some problems

J. Malick, J. Povh, F. Rendl and A. Wiegele
Regularization methods for semidefinite programming
Submitted, 2007

More work:

$\longrightarrow$ to understand the convergence ($\rightarrow$ tuning of parameters)

$\longrightarrow$ to enlarge the range of tackled problems

$\longrightarrow$ to handle inequalities and others cones

(Final) conclusion: SDLS and next

Semidefinite least-squares:
to project onto subsets of SDP cone

$$\text{(SDLS)} \begin{cases} \min & \|X - C\|^2 \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{cases}$$

(Final) conclusion: SDLS and next

Semidefinite least-squares:
to project onto subsets of SDP cone

$$(\text{SDLS}) \begin{cases} \min & \|X - C\|^2 \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{cases}$$

$\longrightarrow$ Efficient solvers for SDLS

(Final) conclusion: SDLS and next

Semidefinite least-squares:
to project onto subsets of SDP cone

$$(\text{SDLS}) \begin{cases} \min & \|X - C\|^2 \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{cases}$$

$\longrightarrow$ Efficient solvers for SDLS

$\longrightarrow$ Besides standard SDP tools, alternative techniques

(Final) conclusion: SDLS and next

Semidefinite least-squares:
to project onto subsets of SDP cone

$$(\text{SDLS}) \left\{ \begin{array}{l} \min \quad \|X - C\|^2 \\ \quad \mathcal{A}X = b \\ \quad X \succcurlyeq 0 \end{array} \right.$$

⟶ Efficient solvers for SDLS

⟶ Besides standard SDP tools, alternative techniques

⟶ Applications (still under development)

1. constructing covariance matrices in finance
2. bounding procedure in combinatorial optimization
3. regularization algorithm for solving SDP

(Final) conclusion: SDLS and next

Semidefinite least-squares:
to project onto subsets of SDP cone

$$(\text{SDLS}) \begin{cases} \min & \|X - C\|^2 \\ & \mathcal{A}X = b \\ & X \succcurlyeq 0 \end{cases}$$

$\longrightarrow$ Efficient solvers for SDLS

$\longrightarrow$ Besides standard SDP tools, alternative techniques

$\longrightarrow$ Applications (still under development)

1. constructing covariance matrices in finance
2. bounding procedure in combinatorial optimization
3. regularization algorithm for solving SDP

thanks !